Universitat Politècnica de Catalunya

ESEIAAT

BACHELOR'S DEGREE THESIS IN AEROSPACE VEHICLES ENGINEERING

Study of the interlink between small satellites in a constellation

REPORT

Author: David Puig Puig
Director: Miquel Sureda Anfrès
Co-director: Jorge Mata Díaz

10th May 2019

"Remember to look up at the stars and not down at your feet. Try to make sense of what you see and wonder about what makes the universe exist. Be curious. And however difficult life may seem, there is always something you can do and succeed at. It matters that you don't just give up."

— STEPHEN HAWKING

Acknowledgements

I would like to express my very great appreciation to Miquel Sureda who, as a director, supervised my project in a fantastic way. Not only for the valuable advice and guideline but also for the encouragement and patience. My special thanks are extended to Jorge Mata. As a co-director he contributed in the guidance of this study.

I also wish to acknowledge the help provided by a previous ESEIAAT student for the "Study of the ground - to - Very Low Earth Orbit (VLEO) satellite communication link", developed by Cànida Muñoz.

I am particularly grateful for the assistance given by the online community. Without them this project would have been impossible. Not only for people who provides solutions but also for those who raise the questions.

Finally, I would like to express my infinite gratitude to my family and friends who, for some reason, always had a great confidence in me.

Contents

List of Figures

List of Tables

List of Abbreviations

ECI Earth Centered Inertial. 37

GEO Geostationary Orbit. 13, 17, 40, 43, 59

GSO Geosynchronous Orbit. 13

HEO High Earth Orbit. 13

ISL Inter-satellite Link. 40, 41

LEO Low Earth Orbit. 2, 13, 17–19, 40, 43, 59, 60

MEO Medium Earth Orbit. 13, 19, 40, 43, 59

NASA National Aeronautics and Space Administration. 30, 40

NORAD North American Aerospace Defence Command. 20, 30, 47

SDP Simplified Deep Space Perturbations. 20

SGP Simplified General Perturbations models. 20

SOC Streets of coverage. 15

TLE Two-Line Element. v, 2, 18–21, 30–32, 47, 49, 51, 55, 58, 59

TT&C Tracking, Telemetry, and Command. 8

UHF Ultra High Frequency. 41

UTC Coordinated Universal Time. 2

VHF Very High Frequency. 41

VLEO Very Low Earth Orbit. 2, 13

List of Symbols

1. Introduction

1.1 Aim

The aim of this study is to obtain a general solution for the rapid determination of visibility windows between a pair of satellites. That will lead to a greater objective, solving the exact same problem for all the satellites in a constellation. To achieve this, the necessary orbital parameters will be evaluated in the most generic way in order to deliver a wide end to end solution.

1.2 Scope

This study will face the following topics:

- Brief study of the basics of orbital mechanics and constellation design.
- Study of the analytic problem defining the visibility between two satellites.
- Parametrization of all the variables involved in the visibility problem and algorithm design to determine visibility status over time.
- Development of a Matlab code applying the step described above.

As far as the deliverables are concerned, this study will give as outcome the following:

- **Report**: refers to the current document and includes all the information regarding the general development of this project. This document will be a guided journey for the understanding of this study. It will be composed of the standard common parts: introduction to the project, research of the state of the art, solution's development, results and conclusions.
- **Annexes**: this document will contain all the Matlab code personally written throughout the study and the required extras necessary to understand the final solution.
- **Budget**: will expose the economic investment necessary to bring this study to live, in terms of quality and time. Basically, it will quantify human resources, workplace and tools.
- **Software User Guide**: the purpose of this document is to let any user know how to use the Matlab code properly.
- **Files**: apart from the documents specified above, this study will deliver some extra files, such as the all the Matlab code itself, including even other functions developed by other contributors.

1.3 Requirements

The following statements will define the required framework to embrace the topics exposed in the scope, at section 1.2:

- The starting point for the determination of the orbits will be a TLE. From them, the orbit will be propagated using a perturbation model, if possible.
- Coordinated Universal Time (UTC) will be adopted as the standard to provide a practical solution for everyone.
- The solution will try to be as general as possible, ensuring it can be valid for all kinds of satellites and all kinds of orbits.
- The minimum number of satellites given to the code will be two.
- The Earth will be considered as a perfect sphere when evaluating the visibility in the algorithm. To take into account buildings, mountains and other objects or effects affecting the visibility, an extra distance to the mean Earth radius will be added. That will lead to always solve the visibility for the worst-case scenario.

1.4 Justification

Since the first achievements of the space race, satellite constellations have been increasing in number, gaining importance due to the whole new set of opportunities that this game-changing perspective brought. They took on relevance with applications such as weather forecasting, navigation, astronomy, radio communications, internet communications, phone communications, Earth observation, scientific research and military.

This study can potentially give a valuable solution for the specific problem of satellite to satellite visibility in any constellation thanks to the general nature taken to approach this topic. This means it can provide solutions for a multi-layer constellation, where satellites at different orbital heights work together for a specific purpose.

The results of this project may be particularly beneficial for the upcoming Low Earth Orbit (LEO) and Very Low Earth Orbit (VLEO) constellations. This kind of solutions usually require a higher number of satellites and involve more difficulties in satellite to satellite visibility, as being closer to the Earth leads to lower visibility windows due to the inherent geometry of this particular case.

For instance, communication and Earth observation oriented constellations can take profit of the shorter distances to Earth and, on the other hand, the lower operation costs related to the lower amount of energy for satellite placement and the capability to reduce satellite's size in certain contexts. This projects are usually in need of low latency, quick data sharing or network traffic load balancing, among others. In such a globalized world, where many solutions rely on almost instant data transfers, the development of this study is well-justified.

Ambitious projects like *Starlink* by *SpaceX* (approved in September 2018), which pretends to provide a new space-based Internet communication system, verify the point exposed above. This constellation is planned to be composed of 12,000 satellites. [1]

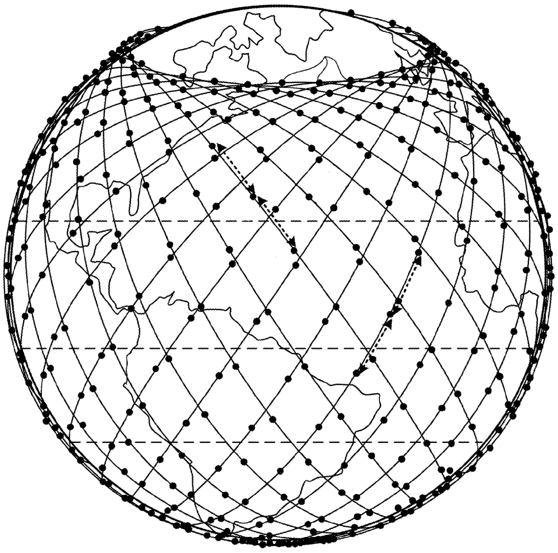


Figure 1.1: Starlink orbital path scheme
[2]

To sum up, the knowledge of the visibility windows between satellites can grant a better performance to ensure fast communications and optimized uplinks and downlinks of information. This better space mission management related to the ability to precisely find out the data transfer windows available will positively impact the power management of every satellite. Basically, they can set power saving modes for the specific times in which the satellites are not supposed to be active.

Moreover, there are other concepts such as fractionated space that should be analyzed (see figure 1.2). A fractionated spacecraft is a space system that distributes its functionalities, such as computation, communication, data storage, payload and even power generation, over several independent satellite modules that share those functionalities through a wireless link. [3] In this frame, *Universitat Politècnica de Catalunya* (UPC) is developing a set of researches in link optics, in which this study is included. Lastly, a mention to the application of the same strategies described previously to other planets should be made. This study would give an interesting approach to this trend also. [4]

Figure 1.2: Fractionated space concept
[5]

1.5 Planning

This section is a roadmap of the different phases that will be encountered during the development of this study. It will specify all the tasks and subtasks necessary in order to achieve the established scope. A Gantt chart will place this tasks in time.

To tackle the software development part, an Agile methodology process will be embraced. Agile rejects sequential phases and relies on simultaneous, incremental work. This will help reducing the risk of being stuck in the development of a specific feature. [6]

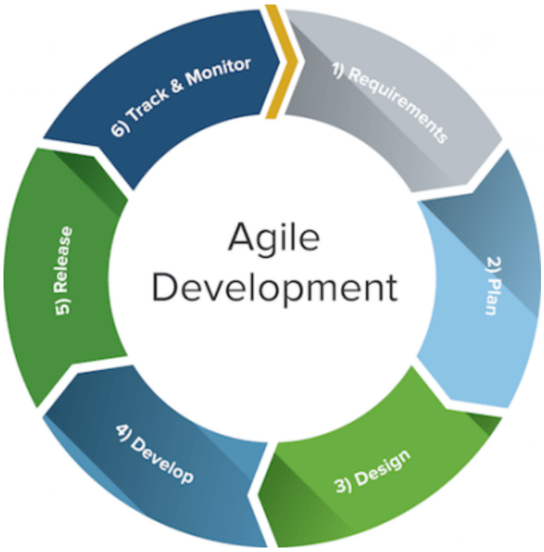


Figure 1.3: Agile lifecycle
[7]

Taking advantage of the flexibility provided by the Agile methodology, it will even be possible to adapt the scope to the difficulties encountered during the development. The software development part, the Matlab code in this case, is identified as the riskiest part in this study. Due to that fact, it will be developed as soon as possible. That will provide valuable information on the exact level of difficulty and will allow the possibility to change the approach before it is too late.

Agile methodology will be applied to the other extensive part of the study, the report. The specific information lacking at this stage of the project will be researched just in time. Part of the information needed to write the report will have been searched before because of the Agile development of the Matlab code. This will help iterating the report to accomplish a better final result.

First, all the tasks accordingly coded for a quick and easy identification will be exposed in the following table.

Table 1.1: Interdependence relationship among tasks

Task code	Task description	Preceding task
0	CONTEXTING	
0.1	Brief initial information research	
0.2	Study of the ground-to-Very Low Earth Orbit (VLEO) satellite communication link	0.1
1	SETTING AN APPROACH	
1.1	Satellite to satellite visibility deep research	0
1.2	Algorithm choice and deep understanding of it	1.1
2	SOFTWARE DEVELOPMENT	1
2.1	Gathering information needed to code	
2.2	Setting GitHub control version	
2.3	Creating code structure	
2.4	Implement Features	
2.5	Menus	
2.6	Physic constants and simulation parameters	
2.7	Main algorithm for two satellites at a given time	
2.8	Input for TLE	
2.9	Logs	
2.10	Main algorithm for two satellites during a time period	
2.11	Main algorithm for a constellation during a time period	
2.12	3D visualization	
2.13	MP4 animation output	
2.14	CSV insertion for data analytics	
2.15	Pathfinding algorithm	
3	VALIDATION	2
3.1	Testing cases	
3.2	Comparison with a professional software	
3.3	Comparison to scientific paper results	
4	REPORT DEVELOPMENT	3
4.1	Deep research on lacking information	
4.1.1	Satellites, orbits and constellations	
4.1.2	Communications architecture	
4.4	Writing of the report	
5	DELIVERABLES	
5.1	Project Charter	1.2
5.2	Follow-up 1	
5.3	Follow-up 2	
5.4	Follow-up 3	
6	CHECKING FINAL DELIVERY	
6.1	Report	4
6.2	Annexes	3
6.3	Budget	4
6.4	Files	3
7	PRESENTATION	
7.1	Presentation development	6
7.2	Demo video	7.1
7.3	Final Presentation	7.2

To conclude this introduction, the tasks described above will be placed in a Gantt chart down below. This first part of the report describes not only what will be developed, but also why. Next chapter will provide a clear context for the complete understanding of the final result. The approach of this project is going to be a reflection of everything defined in this two first chapters.

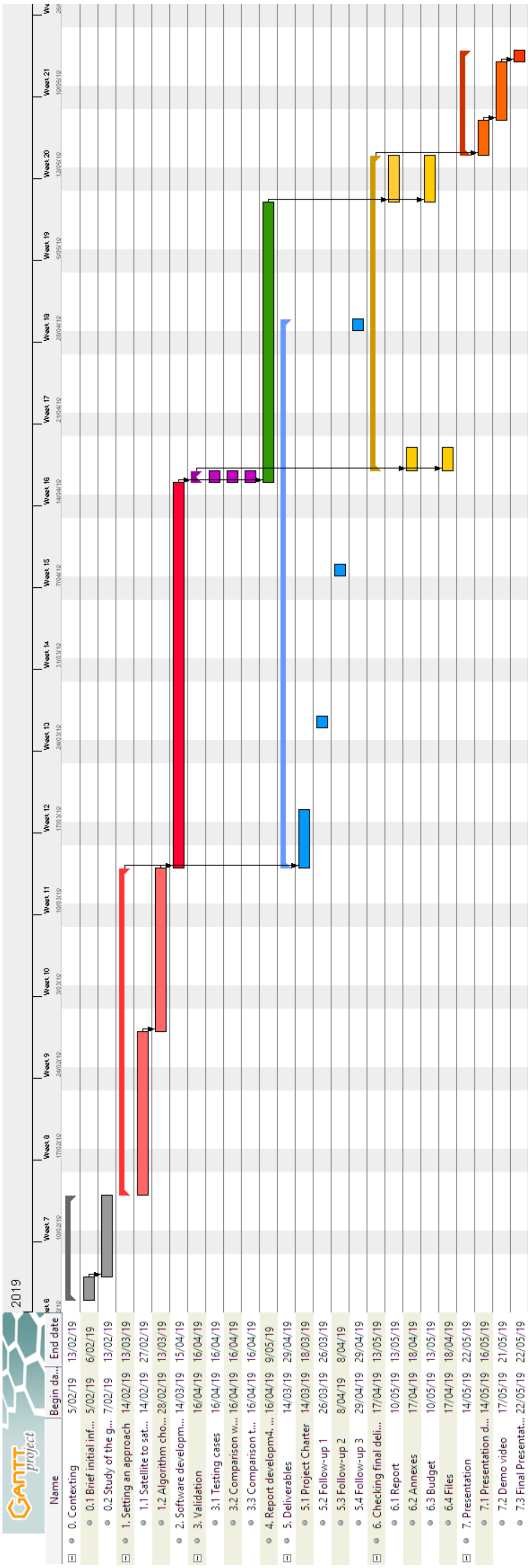


Figure 1.4: Gantt chart

2. State of the art

To understand the upcoming chapters of this study it is essential to explore the history of satellite constellations and, above all, the current state of the art. The subject, from a low-level perspective, are satellites. That is the reason why they will be firstly reviewed. The relationship between them represents the basic actor in the visibility problem. It is mandatory to understand, not only them, but also the orbits in which they move. Then, after studying orbital mechanics basics, constellations will set the high level subject in this project. Eventually, a revision of the last breakthroughs in the satellite to satellite visibility problem itself will close the chapter.

2.1 Satellites

2.1.1 Brief history of satellites

> “Man must rise above the Earth - to the top of the atmosphere and beyond - for only thus will he fully understand the world in which he lives.” - Socrates

Sooner than anyone else, Socrates (Ancient Greek, 470 – 399 BC) predicted the mindset switch that the space industry would trigger. Many years after, with the collaboration of countless genius and dreamers - and some war advances too -, humanity invented the rocket. One able to defeat Earth's gravitational field. And this fact lead, in 1957, to the first artificial satellite to orbit the Earth, the Sputnik 1. It was Russia, and they unleashed something. From then on, everyone set their sights on the sky and, as a consequence 8,378 objects have been launched into space since then. Currently, in 2019, 4,994 are still in orbit. [8]

In this study, a satellite always alludes to an artificial satellite. To give a definition, a satellite is a celestial body or a man-made object orbiting a primary body. Natural and artificial respectively. As said before, the focus is on the ones that mankind deliberately place into orbit.

2.1.2 Satellites description

Satellites can be shaped with regard to their functionality, being suitable for the task that its application demands. This fact does not keep them from sharing some similarities in its construction and elemental parts. The common parts found are: [9]

- **Housing**: it is the structure where all the other systems and subsystems are packed. Its design is clearly influenced by the system employed to stabilize the attitude of the satellite in its orbital slot. Three-axis stabilization or spin stabilization are the two principal approaches to stabilizing attitude control on spacecrafts. The first one can be achieved by small thrusters or electrically powered reaction wheels, second one, on the other hand, is based on rotating the whole spacecraft mass.
- **Power system**: satellites need power to operate, to feed all its systems. The two most common power sources are high performance batteries and solar cells. Solar cells have been improving a lot in its efficiency but due to eclipses, batteries are usually included as a supplemental power source.

- **Antenna system**: designed to provide Tracking, Telemetry, and Command (TT&C) functions to maintain the operation of the satellite in orbit, transmitting and receiving data from mission control. If its application demands it, it will also receive and transmit the telecommunications signals to provide services to its users.
- **Command and Control system**: in charge of monitoring all the vital operating parameters of the satellite and telemetry circuits for relaying this information to the earth station. It is also a system for receiving and interpreting commands sent to the satellite, and a command system for controlling the operation of the satellite.
- **Station keeping**: the mission of this module is the maintenance of a satellite in its assigned orbital slot and in its proper orientation. The physical mechanism for station keeping is the controlled ejection of hydrazine gas from thruster nozzles which protrude from the satellite housing.
- **Transponders**: it is an electronic component that shifts the frequency of an uplink signal and amplifies it for retransmission to the earth in a downlink.
- **Payload**: the last common part are the hardware that is in charge of accomplishing the functionality for which it was designed.

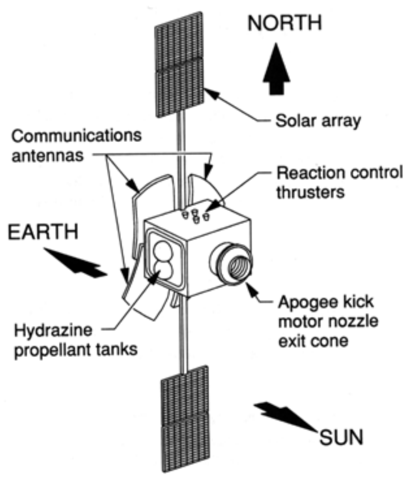


Figure 2.1: Satellite anatomy
[9]

2.1.3 Classifications of satellites

The following classifications in size and application will give an overview of satellites' catalogue.

By size:

- **Femtosatellites**: less than 0.1 kg.
- **Picosatellites**: between 0.1 kg and 1 kg.
- **Nanosatellites**: between 1 kg and 10 kg.
- **Microatellites**: between 10 kg and 100 kg.

- **Minisatellites**: between 100 kg and 500 kg.
- **Medium satellites**: between 500 kg and 1,000 kg.
- **Large satellites**: more than 1,000 kg. The *International Space Station* (ISS) is the largest, with a mass of 419,725 kg. [10]

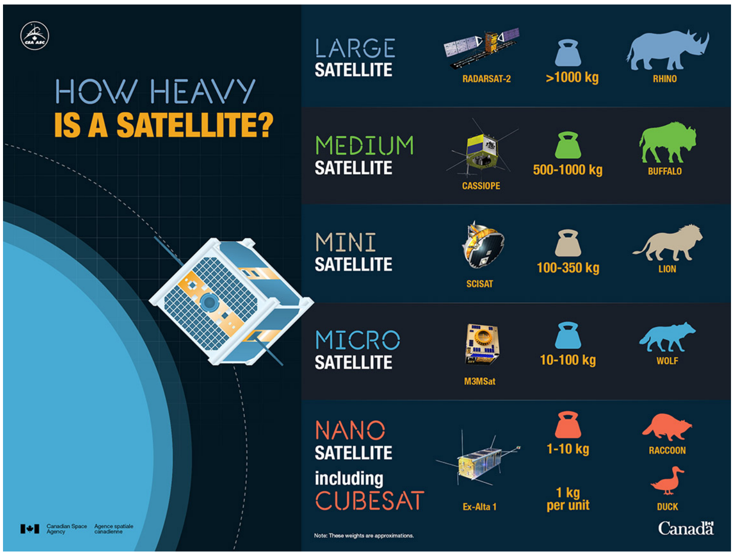


Figure 2.2: Satellite size comparison
[11]

CubeSats deserve a special mention. They are a popular trend, opening new market opportunities to already established companies and start-ups. Its main advantages are: low cost due to less weight and rapid prototyping due to narrow scope. They can be categorized as nanosatellites or microsatellites depending on how many unit modules compose them. Each unit is 10 cm x 10 cm x 11.35 cm and has a weight of 1.33 kg.

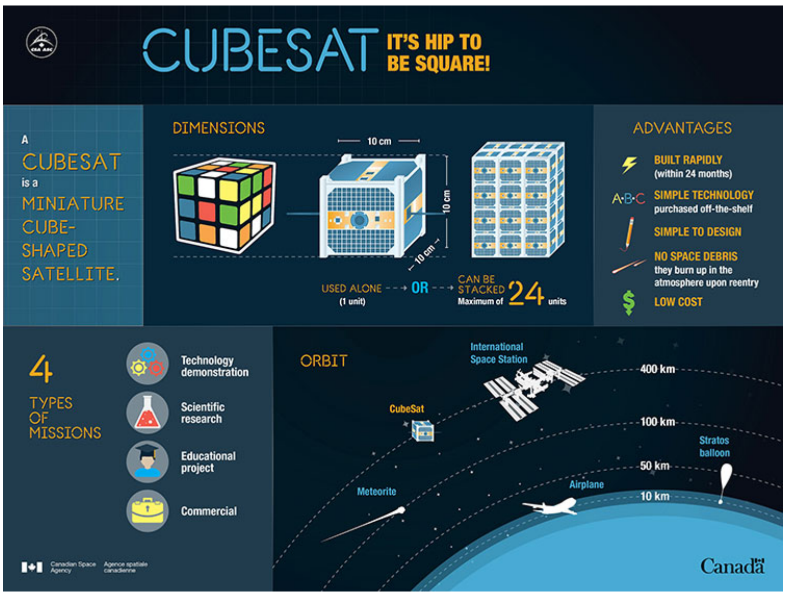


Figure 2.3: *CubeSats* at a glance
[11]

This concept even challenges the standard on how to get to orbit. Using rockets from ground to orbit has been the must during many years. However, alternatives such as carrying a rocket in a plane and launching in the air - “Flying launchpad” - have come to reality by *Virgin Orbit*. [**?**]

Figure 2.4: New spacelaunch methods
[12]

By application:

- **Navigation**: provides accurate position, velocity and time to users. Needs to be a constellation to be 24 hour operational anywhere in the world.
- **Communication**: telecommunications taking advantage of the large territory areas that a satellite or a constellation of satellites can handle. Telephony, television, radio, mobile, internet broadband are some of the specific applications in, perhaps, the most exploded satellites’ utilization.
- **Weather**: analysis of the current state of the atmosphere.
- **Earth Observation**: ability to evaluate Earth changes throughout time is appropriate for agriculture, forestry, geology, risk management, cartography, environment and defense and security. [13]
- **Military**
- **Scientific research**

2.2 Orbits

This study will focus on orbits around the Earth although we have satellites orbiting other planets such as Mercury, Venus, Mars, Jupiter, Saturn and even minor planets, asteroids and comets. *Rosetta* orbiting 67P/Churyumov–Gerasimenko exemplifies it with one of the greatest achievements in space history, sending *Philae* to perform the first successful landing on a comet. Ironically, it would be a mistake to consider this comet a sphere as required in section 1.3 for the development of this study. Nonetheless, applications concerning that kind of celestial objects are not on the horizon.

2.2.1 Brief history of planetary motion

To provide a context, it is mandatory to introduce three modern scientists that made possible the understanding of orbital mechanics. Tycho Brahe (1546–1601) was a Danish astronomer who made exceptionally accurate astronomical and planetary observations. All his work was inherited by Johannes Kepler (1571-1630), also an astronomer, from Germany, who worked as his assistant. All this data helped Kepler define its three laws of planetary motion. They can be described as follows: [14]

1. The path of the planets about the sun is elliptical in shape, with the center of the sun being located at one focus. (The Law of Ellipses)
2. An imaginary line drawn from the center of the sun to the center of the planet will sweep out equal areas in equal intervals of time. (The Law of Equal Areas)
3. The ratio of the squares of the periods of any two planets is equal to the ratio of the cubes of their average distances from the sun. (The Law of Harmonies)

The third one is probably the greatest scientist of all time, Isaac Newton (1642–1727). If Kepler's laws define the motion of the planets, Newton's laws define motion. Down below the most famous three laws in physics, laws of universal gravitation: [15]

1. Every object in a state of uniform motion tends to remain in that state of motion unless an external force is applied to it.
2. The relationship between an object's mass m, its acceleration a, and the applied force F is $F = ma$. Acceleration and force are vectors, in this law the direction of the force vector is the same as the direction of the acceleration vector.
3. For every action there is an equal and opposite reaction.

If any other physicist had to be mentioned, it would be Albert Einstein (1879-1955). He reformulated gravity in its general relativity. However, in order to understand this thesis, notions of relativity are not required at all.

2.2.2 Characterization of an orbit

As it can be expected, orbits have a set of parameters to be uniquely identified. It has to be kept in mind that, at the end, Keplerian orbits are an ideal mathematical approximation of an orbit at a given time. This set of parameters are called orbital elements or Keplerian orbital elements. These are the six above-mentioned parameters:

- **Eccentricity** (e): it defines the shape of the orbit. Actually, it is a measure on how elongated is the orbit compared to a circular orbit. When equal to 0, it means a circular orbit; between 0 and 1, elliptical; 1, parabolic; and more than 1, hyperbolic.

- **Semi-major axis** (a): it has a relation with the size of the orbit. It is the long axis in a elliptical orbit and the radius in a circular orbit. Also defined as the sum of the periapsis (closest point to the orbit from the primary body) and apoapsis (furthest point to the orbit from the primary body) distances divided by two.
- **Inclination** (i): vertical tilt of the ellipse with respect to the reference plane. By this parameters we can characterize equatorial orbits (0º or 180º), polar orbits (90º), direct or prograde (moves in the direction of Earth's rotation) and indirect or retrograde (moves against the direction of Earth's rotation). [16]
- **Longitude of the ascending node** (Ω): horizontal yaw around a perpendicular to the plane of reference. Angle measured from the reference frame's vernal point to the ascending node (where the orbit passes upward through the reference plane)
- **Argument of periapsis** (ω): defines the orientation of the ellipse in the orbital plane, as an angle measured from the ascending node to the periapsis.
- **True anomaly** (f): defines the position of the orbiting body along the ellipse at a specific time.

For a better understanding of the six Keplerian elements see figure 2.5.

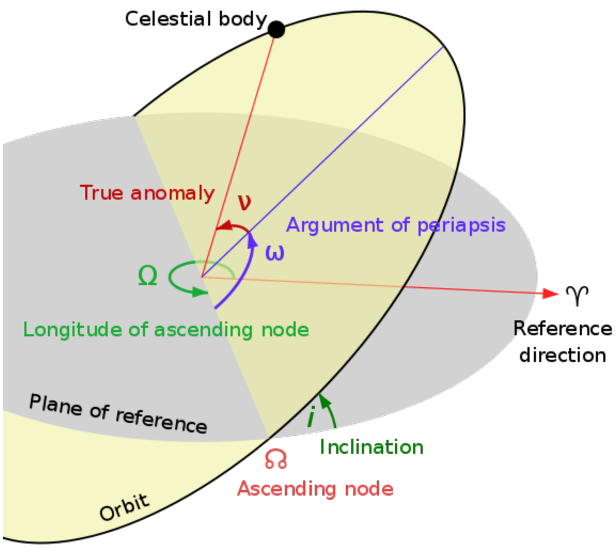


Figure 2.5: Orbital elements
[17]

In addition, other important parameters are:

- **Parameter** or **semi-parameter** (p): distance from the primary focus to the orbit. Typically used to describe the size of a parabolic orbit because the value of the semi-major axis is infinite. [18]
- **Periapsis distance** (q): distance from the primary body to the closest point of the orbit.
- **Mean anomaly** (M): the fraction of an elliptical orbit's period that has elapsed since the orbiting body passed periapsis.

- **Mean motion** (n): angular speed required for a body to complete one orbit, assuming constant speed in a circular orbit which completes in the same time as the variable speed, elliptical orbit.
- **Period** (P): time to complete an orbit - one revolution - around the primary body.
- **Time of perifocal passage** (T): last time the satellite passed through the periapsis. Used as a reference for the time of interest.

2.2.3 Classification of orbits

There exist many kind of orbits - some of them mentioned due to the fact of being related to previously discussed parameters - but table 2.1 shows the classification by orbit height.

This approach is interesting because they change a lot the geometry of the satellite to satellite visibility. VLEO will be considered as a subcategory in LEO. The others will be: Medium Earth Orbit (MEO); Geosynchronous Orbit (GSO) and Geostationary Orbit (GEO), that are orbits around Earth matching Earth's sidereal rotation period but in the case of GEO, the inclination is set to zero; and High Earth Orbit (HEO). HEO should not be confused with highly elliptical orbits (they are sometimes referred as HEO also) like Molniya, which gives complete coverage for high altitudes using a three-satellite constellation.

Table 2.1: Types of orbit by height

Parameter	LEO	MEO	GEO/GSO	HEO
Satellite Height	500-1,500 km	5000-12,000 km	35,800	>35,800
Orbital Period	10-40 minutes	2-8 hours	24 hours	>24 hours
Common Number of Satellites per Constellation	40-80	8-20	3	-
Satellite Life	Short	Long	Long	Long

To end this section, figure 2.6 shows an exaggerated picture of how many satellites are orbiting the Earth. Each point represents a satellite but not at scale. Collisions are extremely rare but it can already be noticed that LEO level is the most crowded one because of the shorter distance. This fact usually implicates larger constellations and, at the same time, it is economically more accessible sometimes.

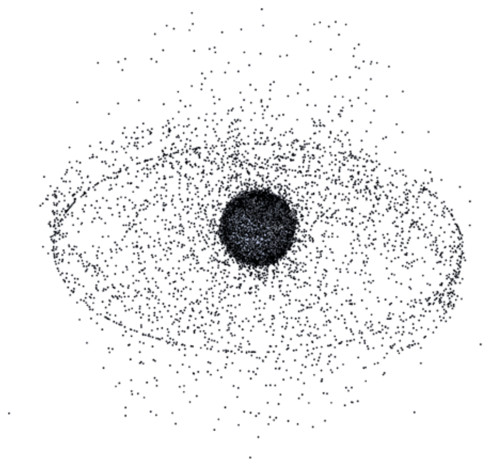


Figure 2.6: Space debris representation by Orbital Debris Program Office [19]

2.3 Satellite constellations

As mentioned in the previous section, there are satellites orbiting other planets but not satellites working as a constellation in other planets. Despite this fact, the design principles of satellites constellations can be applied to other planets and stars. After all, one could wonder where is the limit of gravity when orbiting a primary body. The exception is found while considering a constellation around a black hole, because in that situation, satellite to satellite visibility is completely redefined due to light and electromagnetic waves bending around the black hole. In some way, visibility could be permanent. However, taking into account that the closest black hole is *V616 Mon* at a distance of 3,000 light years, the application in this case is currently unreachable. [20]

2.3.1 Constellation design principles

In previous sections, it has been reviewed how satellites work and perform as individual pieces of hardware. Sometimes it is enough with one satellite to effectuate the desired purpose.

Nonetheless, it is clear that many applications require a group of satellites working as a whole body to accomplish the goal. Satellite constellations are designed to cover these specific applications where having more than a satellite is mandatory. Indeed, this project studies the visibility relationship between two or more satellites and, due to this fact, some research and basic understanding in how constellations are designed is valuable.

In 1960, the US Navy had in mind the purpose of providing assistance to their ballistic missiles. To achieve that they set one of the firsts constellation in history, the *Transit*. Since then, the developments in satellite constellations were propelled by its wide applicability.

There is no specific standard when it comes to constellation design. Some algorithms to meet specific requirements have been developed during the last decades. However, a method to quickly have a solution for every different need is not here, for now. The latest improvements and advances in artificial intelligence could potentially lead to a giant leap forward in this affair. Testings with genetic algorithms corroborate this point of view. [21]

Normally, a constellation is designed to achieve the goal it pursues, minimizing the overall cost of the mission. One might think that this leads to have a constellation with the minimum amount of satellites but the truth is that a higher number of simpler satellites can sometimes lower the cost of the business. For instance, many *CubeSats* can be deployed in the same launch, thus reducing the cost. Besides, as it has already been mentioned, their architecture allow companies to trim the budget.

It is highly convenient to collect and define some parameters to define constellations. Of course, this will help us head to a standardized method. They are presented in table 2.2 down below.

Table 2.2: Parameters to be considered during Constellation Design

Parameters	Mission impacts
Number of satellites	Affects the coverage and the principal cost.
Number of orbital planes	Varies based on coverage needs. Highly advantageous to have minimum number of orbital planes as transfer between the orbits increases the launch, and transfer costs.
Minimum elevation angle	Must be consistent with all satellites. Determines the coverage of single satellite.
Altitude	Increasing the altitude increases the coverage and launch and transfer cost. Decreases the number of satellites. For communication applications, increase/decrease in altitude can correspondingly change latency.
Inclination	Determines the latitude distribution of coverage and selected based on coverage needs.
Plane spacing	Uniform plane spacing results in continuous ground coverage.
Eccentricity	Circular orbits are popular, because then the satellite is at a constant altitude, requiring a constant strength signal to communicate. For some cases, elliptic orbits are chosen where we need satellites to stay over a particular region for longer duration. Tundra and Molniya orbits are two such examples.

[22]

Another fact to take into account is avoidance collision. As shown in figure 2.6, satellites need to share space with each other. In fact, in 2012 it occurred the first ever collision between two intact satellites in orbit. Iridium 33 and Cosmos 2251 collided over Siberia and that, unfortunately, reminded to the space industry that, while improbable, satellites can cross paths in the vast space over Earth. Orbital planes are finite and that showed that in a near future orbits can be too crowded.

Constellations can be designed to meet special requirements but typically, constellations are designed towards some Earth-based user application. That means, each satellite can cover part of the Earth surface, for example, in communication satellites. This is called coverage and as one can guess, it increments with the increase in number of satellites or with the increase in altitude. Nonetheless, this also increments the overall satellite development and launch costs.

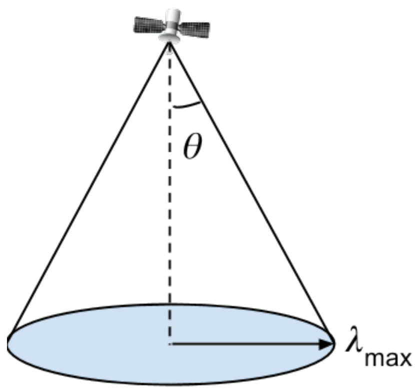


Figure 2.7: Surface coverage for a satellite

[22]

Two main constellation design methods, trying to maximize the coverage optimizing the number of satellites, can be analyzed. Streets of coverage (SOC) and Walker-Delta patterns.

Figure 2.7 above presents the ground trace projected by a satellite. "The ground trace (shaded area) is circular with radius λ_{max} and is subtended by a cone with half angle θ. The continuous coverage often called the street of coverage is represented by considering a chordal range of λ_{street} on both sides of the ground trace (assumed circular), as shown in figure below. The adjacent orbits should be decided such that the bulges of one orbital plane fills the dips of the other orbital plane." (Raja P, 2015) [22] Then, the maximum distance D_{max} between contiguous street orbits that ensures perfect coverage can obtained through the following equation:

$$D_{max} = \lambda_{street} + \lambda_{max}$$

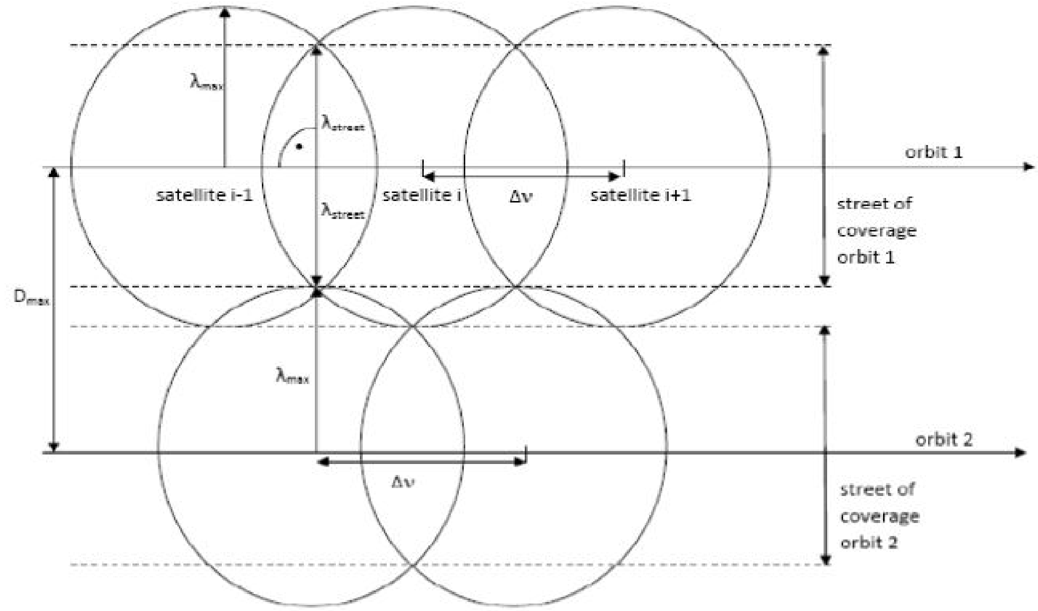


Figure 2.8: Streets of coverage pattern
[22]

"A frequently used design technique is the Walker-Delta pattern constellation for a global coverage of the Earth's surface by a minimum number of satellites in circular orbits. The Walker constellation is denoted by a notation (i: t/p/f). Where *i* means inclination; *t*, total number of satellites; *p*, number of equally spaced orbit planes; and *f*, relative phase difference between satellites in adjacent planes.

A Walker-Delta pattern contains of total of t satellites in p orbital planes with $s = \frac{t}{p}$ satellites in each orbital plane. All orbital planes are assumed to be in same inclination i with reference to the equator. The phase difference between satellites in adjacent plane is defined as the angle in the direction of motion from the ascending node to the nearest satellite at a time when a satellite in the next most westerly plane is at its ascending node. This is illustrated in figure 2.9 below.

In order for all of the orbit planes to have the same phase difference with each other, the phase difference between adjacent satellites must be a multiple f of $\frac{360°}{t}$, where f can be an integer between 0 to $p-1$." (Raja P, 2015) [22] Galileo constellation is an example of a constellation designed by this method.

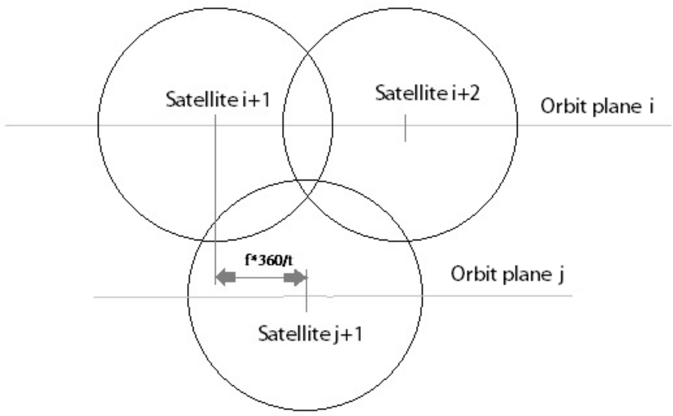


Figure 2.9: Walker-Delta coverage pattern
[22]

Other mentions of interest would be Draim's method (elliptical orbits) or hybrid constellations, having such a difference between orbital planes that it is basically a multi-layer constellation, combining, for example, LEO and GEO satellites.

2.3.2 LEO constellations

As introduced in section 1.3, this study could potentially benefit the design and management of LEO constellations by cause of the inherent geometry implicating shorter visibility windows or even permanent non-visibility.

Geostationary satellites can cover Earth easily due to the fact of being far form Earth. In LEO satellites, many more satellites are required to grant continuous coverage over an area. The amount of satellites combined with lower distances to Earth, turn this types of constellations into a valuable option for many applications. The advantages are the following:

- Lower power needed for data transmission and instrumentation.
- Better resolution for Earth observation applications.
- Low latency thanks to lower distances between satellites and users.
- Cost per satellite is lower as lower distance usually imply smaller and lower-cost satellites compared to GEO satellites, for example. Launch to orbit cost is also lower due to lower orbital height. Nonetheless, many more satellites are needed so, as always, it is worth analyzing the specifications of the constellation.

On the other hand, this are the main downsides:

- High velocity relative to surface imply short contact periods. Reason why ground to satellite and satellite to satellite visibility windows analysis is key.
- Atmospheric drag effects impact the lifespan of satellites in LEO orbits as they require regular maintenance to keep them on track.

- Efficiency is not good since small satellites in LEO orbits spend most of their orbit over oceans and other unpopulated areas. This geographical inefficiency means that much more capacity must be placed into LEO orbit than what is actually needed to provide service. [23]

One typical example for LEO constellations is *Iridium*, a 66 communication satellites constellation arranged in 6 LEO orbital planes. It was designed using Walker-Delta approach.

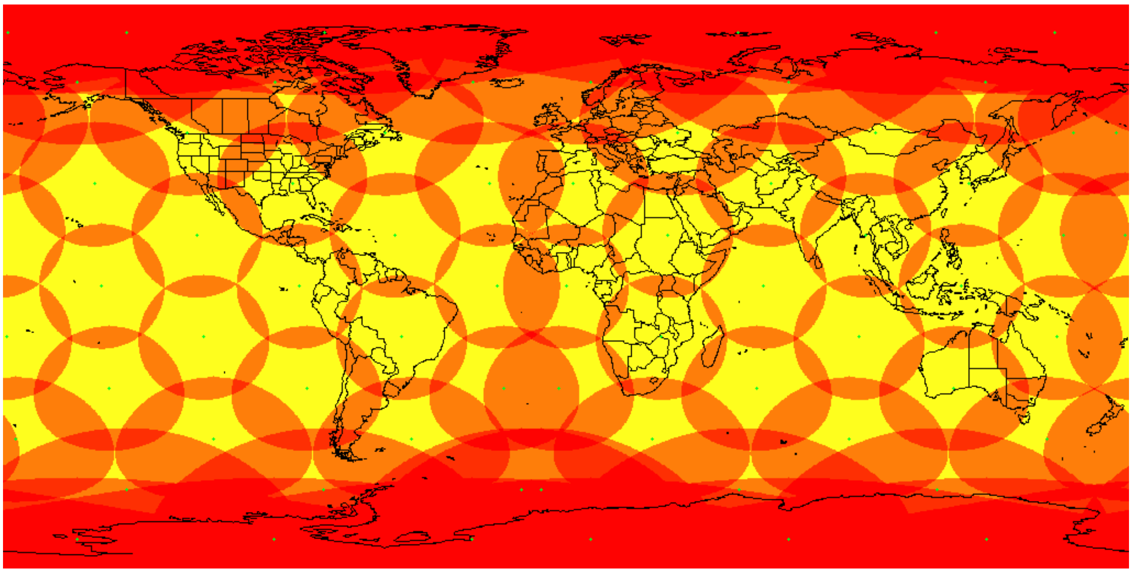


Figure 2.10: *Iridium* constellation coverage
[24]

2.4 Visibility problem

This final section of the state of the art presents the different particularities arising from the conceptual problem of the determination of the visibility in relation to satellites. Basically, the interest is on two clear cases regarding the visibility: ground station to satellite analysis and satellite to satellite analysis. Visibility is in both cases defined as the possibility to trace a straight line between the two subjects of study, either if it is between to satellites or a ground station and a satellite.

Note that this problem is always solved per pairs. It is not dependant on if it is a ground to satellite or a satellite to satellite analysis and it doesn't matter if the analysis takes into account more satellites or more ground stations. It will always be needed to solve the two subject problem to give an answer to the problem involving more subjects of study.

It has to be kept in mind that solving the problem over time involves a satellite tracking module. The main goal of the study is to provide an answer to the visibility problem at a given time. Thus, getting the parameters for the next calculation in time - where the two subjects of study will have moved and changed in some of its parameters - will be the only lacking piece to obtain the truly powerful result, calculating visibility windows in a time period. In order to track satellites, and as a result, solve the next time step, two feasible options are on the table:

- **Stationary orbit**: TLE data will define the orbit. In any case, at any time, the orbit will remain the same as defined by the orbital elements extracted from the TLE. The satellite will cover this orbit as its period and mean motion describe.

- **Orbit propagation**: the parameters needed to solve the visibility problem will emanate from the TLE but a perturbations model will handle to give accurate parameters over time. Perturbations models take into account physical phenomena such as Earth's oblateness and atmospheric drag.

2.4.1 Orbit perturbations

Perturbations can come from Earth's oblateness, atmospheric drag, third-body effects, solar wind or radiation pressure and electromagnetic drag (see table 2.3 to have some notions of its consequences).

Earth's oblateness alludes to the fact of the Earth not being a perfect sphere. Earth is squashed, the north pole is more pointed than the flatter south polar region and Equator is an ellipse if it is observed from the top. Mass is not distributed evenly, thus affecting the Earth's gravitational field. This effect is also usually called as J2 effect, which is the predominating coefficient coming from the spherical harmonic functions bonded to the sphere deviations. Gains importance in LEO and MEO orbits. [25]

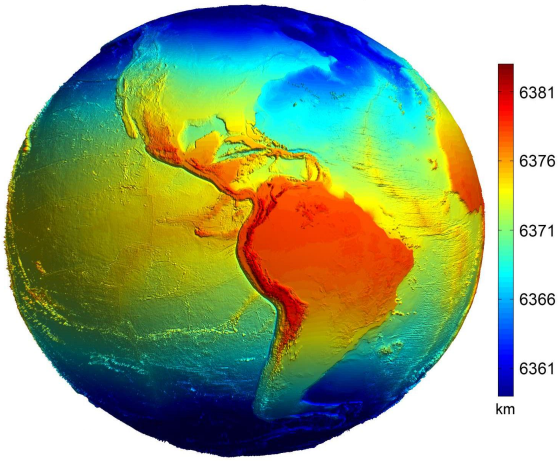


Figure 2.11: Earth oblate spheroid with distances from the geocenter represented in color [26]

The earth's atmosphere trails off into space, being less dense but still containing particles and that implies having atmospheric drag, again, very important in LEO orbits. It can lead to orbital decay, phenomena of pushing satellites into a lower orbit and, eventually, hitting the primary body surface. Lower orbits mean more drag and more drag means being pushed to a lower orbit. So, as it can be noticed, it is something to be mindful of to ensure that the mission for a satellite or constellation lasts as long as it was initially planned.

Third-body effects are a consequence of the physical impossibility of having an ideal binary system in universe. The Sun, the moon and even Jupiter affect satellites' orbit, particularly in high orbits, in this case. This study considers a binary system - satellite and primary body - for the approach, equations and algorithm.

Radiation pressure phenomena is caused by the Sun. These constantly expelled particles speed up and slow down satellites, becoming one of the other types of orbital perturbations.

Electromagnetic drag appears because of the own magnetic field created by the electronic components and interacting with Earth's magnetic field, causing mainly torque on satellites. The effect on the orbit is not noticeable.

To predict more accurately the position and velocity of a satellite, mathematical models were created. These models take into account the effects which deviate the satellite from its ideal orbit, set by a TLE in this study. Models are divided by the ones dedicated to deep space and the general ones, Simplified Deep Space Perturbations (SDP) and Simplified General Perturbations models (SGP), respectively. Down below, a description for the propagators that have been created and updated since the sixties: [27]

- **SGP** was the first orbit propagator. Developed by Hilton and Kuhlman in 1966 thanks to Kozai research made in 1959. Suitable for satellites orbiting near the Earth as it considers satellites with an orbital period lower than 225 minutes. This model assumes low eccentricity and constant perigee's altitude.
- **SGP4** was later developed by Ken Cranford in 1970. It is an improvement of the previous propagator in order to track the growing number of satellites in orbit. It is also mainly used for near Earth satellites.
- **SDP4**, developed by Hujsak in 1979, is the SGP4 propagator adapted for deep space objects. This considers satellites with an orbital period greater than 225 minutes. For periods above this value, the satellite's orbit is perturbed by the moon and the sun but also by some resonance effects in periods of 12 and 24 hours.
- **SGP8**, also used for near Earth satellites, is almost like the SGP4 propagator but the calculation methods are different. However, it follows the same models for the atmospheric and gravitational effects.
- **SDP8** is the SGP8 propagator adapted to deep space effects. In addition, SGP8 and SDP8 are better at managing orbital decay.
- **HPOP** propagator, which stands for High Precision Orbit Propagator, is the most accurate of all the above-mentioned models. It gives an error that has an order of only 12 meters per orbit. This propagator takes into account perturbations such as gravitational effects of the moon and the sun, atmospheric drag and solar radiation pressure. The North American Aerospace Defence Command (NORAD) element sets are provided using SGP4 or SDP4. They make precise observations and through some mean values TLE sets are finally obtained. Therefore, it is more consistent to implement. [28]

Table 2.3: Comparison between perturbations effects

Perturbation	Acceleration m/s^2	Orbital effect in 3 hours	in 3 days
Central force (as a refernce)	0.56		
J2	5.1e-05	2 km	14 km
Rest of harmonics	3.1e-07	50-80 km	100-1,500 m
Solar + moon gravity	5.1e-6	5-150 km	1,000-3,000 m
Tidal effects	1.1e-9	-	0.5-1 m
Solar radiation pressure	1.1e-7	5-10 km	100-800 m

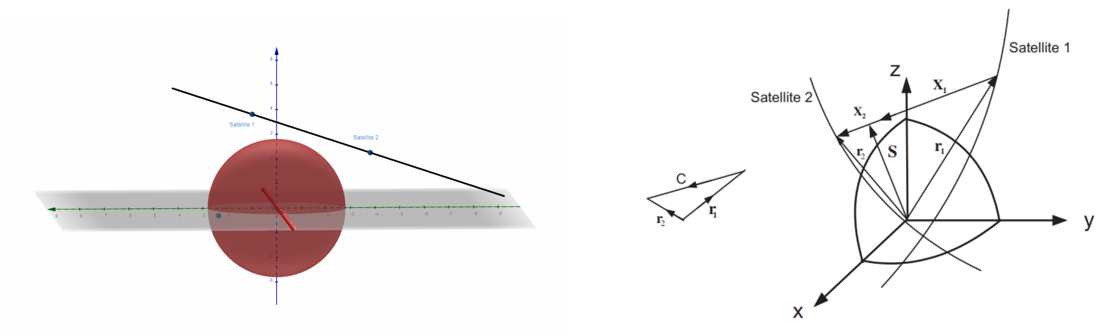
[29]

2.4.2 Selected approach

The ground to satellite visibility problem is widely developed in the scientific literature. That's the reason why professionals and enthusiasts have created different computer codes, in different programming languages, with different approaches, to tackle that problem.

On the other hand, satellite to satellite visibility is not that well-developed. The lack of scientific papers drives to a lack of codes. With *FreeFlyer* software a satellite to satellite visibility analysis can be executed.

As it will be discussed in the next chapter, a geometric approach is the way to solve this problem. One first thought was to take it as a 3D geometry problem where Earth would be a sphere and a line from satellite to satellite - at a given moment - would be drawn. Then, solving the intersection problem the visibility is supposed to be determined; intersection means non-visibility and no intersection means direct line of sight. This alternative is not the path followed in the thesis. The final approach is similar, but, as it is explained in the next chapter, science literature embraces an approach of tracing a line between satellites and, a perpendicular to this one passing through the center of the Earth, or whatever primary body of study.



(a) Intersection approach

(b) Perpendicular line approach

Figure 2.12: Considered approaches
[30]

Synthesizing the previous considerations, the basic rules defining the approach of the satellite to satellite visibility problem for the present study can be stated:

1. **Direct line of sight** will be defined as the possibility to trace a straight line from satellite to satellite. The impossibility will define **non-visibility**.
2. It will always be resolved per **pairs of satellites**.
3. The input information to feed the algorithm will be a **set of TLE**.
4. To analyze visibility windows between satellites, **tracking satellite orbits** is a must, propagating it or not.
5. **Earth rotation** cannot affect the geometry.

3. Interlink architecture

This chapter contains the problem-solving process for the geometry involved in the visibility between satellites. After that, the algorithm is applied in a Matlab code in order to have a useful, revealing and quick tool to analyze visibility windows in a constellation of satellites.

3.1 Analytic approach

3.1.1 Geometry analysis

This section is based on the work presented by M.A. Sharaf, M.E. Awad, I.A. Hassan, R. Ghoneim and W.N. Ahmed in the article “Visual contact for two satellites orbits under J2-gravity” in 2012. [30]

As it was defined in the rules presented in section 2.4.2, the three dimensional satellite to satellite visibility problem can be solved as a two dimensional problem implicating the plane characterized by the two satellite, considered as points, and the Earth’s centre.

This configuration leads to an easy understanding way to find a parameter that tells if there is visibility between two satellites or not. Basically, the idea consists in tracing a line between satellites and the perpendicular to this line that passes through the center of the Earth. Now, if this last mentioned line is greater than the Earth’s radius plus the extra distance set as a tolerance, there is visibility between satellites. If the contrary happens, there is no visibility. Through vector analysis this problem can be solved.

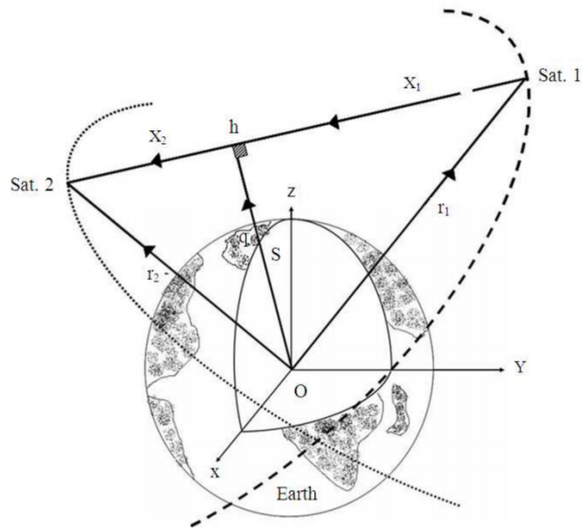


Figure 3.1: Direct line of sight between to satellites
[31]

Remember that Earth is considered a perfect sphere - means Earth radius (a_e) is constant - and an extra distance (Δ) set takes into account not only, buildings and mountains but also atmosphere effects. Indeed, setting an extra distance that takes into account atmosphere effects exceeds any building and any mountain. The magnitude of this vector reflects down below this idea:

$$S = a_e + \Delta \tag{3.1}$$

In fact, the reasoning can be simplified by considering a rise-set configuration. Let's define rise or set between satellites as the very first or last moment visual contact between two given satellites happens. Starting the analysis in this configuration will lead to an equation that sets an equilibrium between visibility and non-visibility. Then, depending on the real configuration, a negative value representing visibility, or a positive value representing the opposite, will be obtained.

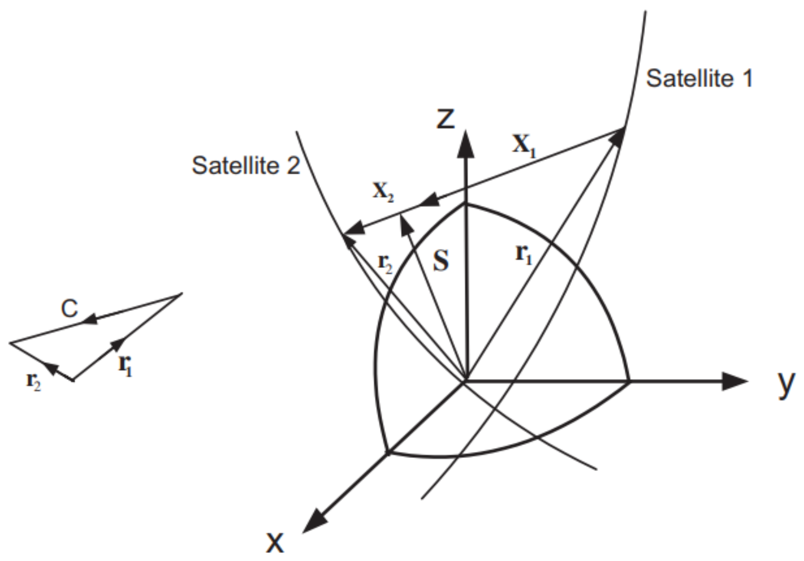


Figure 3.2: Relative rise-set geometry
[30]

Observing the configuration set in figure 3.2 it is possible to define some fundamental vectors, the ones that define satellites' position:

$$\vec{r}_1 = \vec{S} - \vec{X}_1 \tag{3.2}$$

$$\vec{r}_2 = \vec{S} + \vec{X}_2 \tag{3.3}$$

For i: 1,2 referring to each satellite, $\vec{r}_i$ is the position in space and, as it can be seen, this positions are related to $\vec{S}$ and the vectors from $\vec{S}$ to each satellite, $\vec{X}_i$.

Another condition that is always true is that $\vec{S}$ will always be perpendicular to the line or vector $\vec{C}$ traced between both satellites.

$$\vec{S} \cdot \vec{X}_1 = \vec{S} \cdot \vec{X}_2 = 0 \tag{3.4}$$

Still on the subject, vector $\vec{C}$ can be defined as follows:

$$\vec{C} = \vec{r}_2 - \vec{r}_1 \tag{3.5}$$

$$C = X_1 + X_2 \tag{3.6}$$

$$C = \sqrt{(\vec{r}_2 - \vec{r}_1) \cdot (\vec{r}_2 - \vec{r}_1)} \tag{3.7}$$

Using law of cosines it can be expressed in a more handy way:

$$C = \sqrt{r_1^2 + r_2^2 - 2(\vec{r}_1 \cdot \vec{r}_2)} \tag{3.8}$$

Using 3.2 and 3.3 or applying the Pythagorean theorem to the right-angled triangles:

$$X_1 = \sqrt{r_1 - S} \tag{3.9}$$

$$X_2 = \sqrt{r_2 - S} \tag{3.10}$$

With equations 3.9 and 3.10 into 3.6:

$$C = \sqrt{r_1 - S} + \sqrt{r_2 - S} \tag{3.11}$$

As we already know the module of vector $\vec{C}$ it is possible to make the substitution from 3.8 to 3.11:

$$\sqrt{r_1^2 + r_2^2 - 2(\vec{r}_1 \cdot \vec{r}_2)} = \sqrt{r_1 - S} + \sqrt{r_2 - S} \tag{3.12}$$

Then, squaring twice and reordering:

$$(\vec{r}_1 \cdot \vec{r}_2)^2 - r_2^2 r_1^2 + (r_2^2 + r_1^2) S^2 - 2S^2 (\vec{r}_1 \cdot \vec{r}_2) = 0 \tag{3.13}$$

Note that the previous equation forces the rise-set configuration because this was the defined approach. S is known and the input are r_i, but it is not mandatory to fulfill a rise-set configuration. Placing a parameter R allows us to tell if there is visibility or not.

Finally, the visibility equation:

$$R = (\vec{r}_1 \cdot \vec{r}_2)^2 - r_2^2 r_1^2 + (r_2^2 + r_1^2) S^2 - 2S^2 (\vec{r}_1 \cdot \vec{r}_2) \tag{3.14}$$

With R defining the visibility in the cases presented hereunder:

- R < 0 Direct line of sight
- R = 0 Relative rise or set between satellites
- R > 0 Non-visibility

Considering equation 3.13 again, a trick can be done to obtain more information. Let's think that the approach taken is exactly the same except that a rise-set condition is not forced. Then, the equation can be set to be equal to zero as it was initially set, r_i can be the input and S the unknown value. It can be noticed that, then, S will just be the distance from the Earth's centre to the line traced between satellites. So removing Earth's radius from it, this value will give the margin (R_v) between Earth's surface and the satellite to satellite line. In this case, it's obvious that will be positive for direct line of sight and negative for non-visibility case. However, it can be demonstrated by setting an extreme situation.

Clearing S from equation 3.13:

$$S = \sqrt{\frac{r_2^2 r_1^2 - (\vec{r}_1 \cdot \vec{r}_2)}{r_2^2 + r_1^2 S^2 - 2(\vec{r}_1 \cdot \vec{r}_2}} \tag{3.15}$$

As it was expressed before, $S = R_v + a_e$ where R_v is the margin between surface and satellites line and a_e is Earth's radius.

$$R_v = \sqrt{\frac{r_2^2 r_1^2 - (\vec{r}_1 \cdot \vec{r}_2)}{r_2^2 + r_1^2 S^2 - 2(\vec{r}_1 \cdot \vec{r}_2}} - a_e \tag{3.16}$$

To prove that positive values will set visibility in this particular case, a non-visibility case will be analyzed. In figure 3.3 it can be seen that visibility is not possible. Hence, let's see the value for the margin.

$$\vec{r}_1 \cdot \vec{r}_2 = r_1 r_2 cos(\pi) = -r_1 r_2$$
$$R_v = -a_e$$

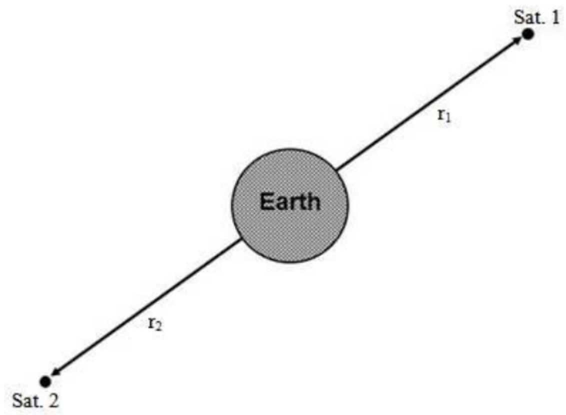


Figure 3.3: Non-visibility geometry
[31]

Note that in this case a_e could be $a_e + \Delta$ to take into account the extra distance. Then, R_v is defining the visibility as presented down below:

- $R_v > 0$ Direct line of sight

- $R_v = 0$ Relative rise or set between satellites
- $R_v < 0$ Non-visibility

3.1.2 Function parameter reduction

The distance between Earth's center and a satellite can be expressed in terms of eccentricity (e), semi-parameter (p) and true anomaly (f).

$$r_i = \frac{p_i}{1 + e_i cos(f_i)} \tag{3.17}$$

What needs to be done is a transformation for both satellites to a reference in which $\vec{r}_i$ can be consistent. Value r_i obtained in terms of orbital parameters in the previous result can be projected to the perigee line and its perpendicular (node line if it has inclination). Using f_i it is possible to do that:

$$\xi_i = r_i cos(f_i) \tag{3.18}$$

$$\eta_i = r_i sin(f_i) \tag{3.19}$$

Down below are presented the standard orientation vectors P and Q, where P is a unit vector from the dynamical center which points at perigee of the orbit and Q is advanced to P by a right angle in the plane and direction of motion. As introduced in section 2.2.2; ω is the argument of the perigee, Ω is the longitude of the ascending node and I is the inclination. This parameters will be the Euler angles and the transformation will bring the position from the orbital plane to the inertial frame established in the equatorial plane of the central body.

$$P_{x_i} = cos(\omega_i)cos(\Omega_i) - sin(\omega_i)sin(\Omega_i)cos(I_i) \tag{3.20}$$

$$P_{y_i} = cos(\omega_i)sin(\Omega_i) + sin(\omega_i)cos(\Omega_i)cos(I_i) \tag{3.21}$$

$$P_{z_i} = sin(\omega_i)sin(\Omega_i) \tag{3.22}$$

$$Q_{x_i} = -sin(\omega_i)cos(\Omega_i) + cos(\omega_i)sin(\Omega_i)cos(I_i) \tag{3.23}$$

$$Q_{y_i} = -sin(\omega_i)sin(\Omega_i) + cos(\omega_i)cos(\Omega_i)cos(I_i) \tag{3.24}$$

$$Q_{z_i} = cos(\omega_i)sin(\Omega_i) \tag{3.25}$$

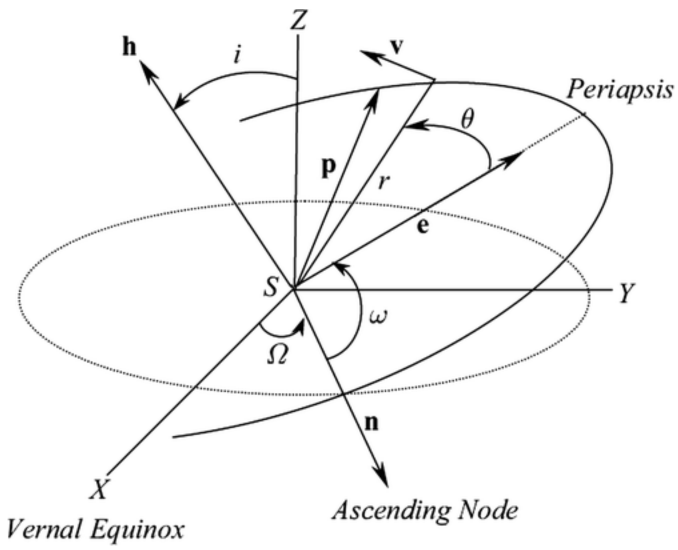


Figure 3.4: Euler angle transformation
[32]

Then the position vector for each satellite can be expressed as:

$$\vec{r}_i = \xi_i \vec{P}_i + \eta_i \vec{Q}_i \tag{3.26}$$

The only thing it is necessary is the scalar product between the two satellite position vectors. It follows as:

$$\vec{r}_1 \cdot \vec{r}_2 = (\xi_1 \vec{P}_1 + \eta_1 \vec{Q}_1) \cdot (\xi_2 \vec{P}_2 + \eta_2 \vec{Q}_2) \tag{3.27}$$

Now it can be further developed:

$$\vec{r}_1 \cdot \vec{r}_2 = (\vec{P}_1 \cdot \vec{P}_2)\xi_1\xi_2 + (\vec{P}_1 \cdot \vec{Q}_2)\xi_1\eta_2 + (\vec{Q}_1 \cdot \vec{P}_2)\eta_1\xi_2 + (\vec{Q}_1 \cdot \vec{Q}_2)\eta_1\eta_2 \tag{3.28}$$

With $A_1 = \vec{P}_1 \cdot \vec{P}_2$; $A_2 = \vec{Q}_1 \cdot \vec{P}_2$; $A_3 = \vec{P}_1 \cdot \vec{Q}_2$; $A_4 = \vec{Q}_1 \cdot \vec{Q}_2$

$$\begin{aligned} \vec{r}_1 \cdot \vec{r}_2 = A_1 r_1 r_2 cos(f_1) cos(f_2) + A_3 r_1 r_2 cos(f_1) sin(f_2) + \\ + A_2 r_1 r_2 sin(f_1) cos(f_2) + A_4 r_1 r_2 sin(f_1) sin(f_2) \end{aligned} \tag{3.29}$$

Substituting r_i from equation 3.17:

$$\begin{aligned} \vec{r}_1 \cdot \vec{r}_2 = \frac{p_1 p_2}{[1 + e_1 cos(f_1)][1 + e_2 cos(f_2)]} \big[A_1 cos(f_1) cos(f_2) + A_3 cos(f_1) sin(f_2) \\ + A_2 sin(f_1) cos(f_2) + A_4 sin(f_1) sin(f_2) \big] \end{aligned} \tag{3.30}$$

Defining some parameters it is possible to follow and get to the reduced parameter solution.

$sin(\gamma) = \frac{A_2}{\sqrt{A_1^2 + A_2^2}}$; $cos(\gamma) = \frac{A_1}{\sqrt{A_1^2 + A_2^2}}$; $sin(\psi) = \frac{A_4}{\sqrt{A_3^2 + A_4^2}}$; $cos(\psi) = \frac{A_3}{\sqrt{A_3^2 + A_4^2}}$

$D_1 = \sqrt{A_1^2 + A_2^2}$; $D_2 = \sqrt{A_3^2 + A_4^2}$;

Finally the scalar product and the equation desired is as follows:

$$\vec{r}_1 \cdot \vec{r}_2 = \frac{p_1 p_2 [D_1 cos(f_2) cos(\gamma_1 - f_1) + D_2 sin(f_2) cos(\psi_1 - f_1)]}{[1 + e_1 cos(f_1)][1 + e_2 cos(f_2)]} \tag{3.31}$$

$$\begin{aligned} R = p_1^2 p_2^2 [D_1 cos(f_2) cos(\gamma_1 - f_1) + D_2 sin(f_2) cos(\psi_1 - f_1)]^2 \\ -p_1^2 p_2^2 + S^2 \left[p_1^2 [1 + e_2 cos(f_2)]^2 + p_2^2 [1 + e_1 cos(f_1)]^2 \right] \\ -2S^2 p_1 p_2 [D_1 cos(f_2) cos(\gamma_1 - f_1) + D_2 sin(f_2) cos(\psi_1 - f_1)] \\ \cdot [1 + e_1 cos(f_1)][1 + e_2 cos(f_2)] \end{aligned} \tag{3.32}$$

Notice that Earth's rotation cannot affect the geometry analysis.

3.1.3 Basic algorithm

The inputs needed are the following:

- Mean motion (n_i^*)
- Mean anomaly (M_i^*)
- Eccentricity (e_i^*)
- Inclination (I_i^*)
- Argument of the perigee (ω_i^*)
- Longitude of the ascending node (Ω_i^*)
- Epoch time (t_i^*)
- Earth radius (a_e)
- Extra distance (Δ)
- Simulation time (t)
- Standard gravitational parameter (μ)

The (*) inputs refer to parameters that are known for the satellite at a given epoch time (t_i^*). The analysis can be conducted at another time - simulation time (t) - denoted by any symbol. As the analysis is always between only two satellites, i=1,2.

This are the steps needed to solve the visibility problem at a given time:

1. Finding the time of perifocal passage T.

$$M_i^* = n_i^* (t_i^* - T_i) \rightarrow T_i = t_i^* - \frac{M_i^*}{n_i^*}$$

2. Finding mean anomaly at simulation time.

$$M_i = n_i(t - T_i)$$

3. Finding true anomaly using Newton-Raphson iteration method.
   a) $E_{i_0} = M_i$
   b) $E_{i_{n+1}} = E_{i_n} + \frac{M_i + e_i sinh(E_{i_n} - E_{i_n})}{1 - e_i cosh(E_{i_n})}$
   c) $E_{i_n} = E_{i_{n+1}}$
   d) If $|E_{i_{n+1}} - E_{i_n}| > 10^{-8}$ go to b. Else $E_i = E_{i_n}$
   e) $f_i = \arctan\left(\frac{sin(E_i)\sqrt{1-e_i^2}}{cos(E_i) - e_i}\right)$
   d) End.

4. Finding distance r_i from Earth's center to satellite.
   First, semi-major axis is found:

$$a_i = \left(\frac{\mu}{n_i^2}\right)^{\frac{1}{3}}$$

   Second, periapsis distance is found:

$$q_i = a_i \cdot (1 - e_i)$$

   Finally,

$$r_i = \frac{q_i(1 + e_i)}{1 + e_i cos(f_i)}$$

5. Finding $\vec{P}_i$ and $\vec{Q}_i$ from equation 3.20.

6. Determine visibility in equation 3.32. Negative R means visibility, positive R means non-visibility.

7. End.

4. Interlink code

This section presents the basic modules implemented in Matlab to determine the visibility windows between satellites in a constellation. The whole code is included in *Annexes* document. Hence, only the key modules are fully included and discussed hereunder.

This code follows the arguments reviewed in section 3.1. The explanation of the code will be led by the complexity, starting with the basic algorithm between two satellites at a given time and ending for a constellation during a period.

4.1 Input parameters

The basic input parameters can be divided in two main areas: common parameters or constants and satellite orbital parameters. As far as the common parameters are concerned:

- Earth radius (a_e): set to 6,371 km.
- Extra distance (Δ): set to 60 km.
- Simulation time (t): anytime expressed as UNIX[1] UTC time.
- Standard gravitational parameter (μ): $3.986 \cdot 10^{14} \left[\frac{m^3}{s^2}\right]$ for the Earth.

On the other hand, data related to each satellite (i=1,2) is:

- Mean motion (n_i^*)
- Mean anomaly (M_i^*)
- Eccentricity (e_i^*)
- Inclination (I_i^*)
- Argument of the perigee (ω_i^*)
- Longitude of the ascending node (Ω_i^*)
- Epoch time (t_i^*)

As it can be noticed, common parameters are already known because they are defined by the celestial object (Earth), or chosen by the user (time and extra distance). Next step is to have a form to obtain satellites' data. As described in the requirements at section 1.3, TLE will give this starting point.

Two-line elements are a set of orbital elements and other data related to an Earth-orbiting object at an epoch time that. This data is encoded in text files. Originally encoded in punch cards back in the seventies by the same providers than today, the NORAD and the National Aeronautics and Space Administration (NASA). They provide this sets through observation and mean values using SGP4 perturbation model.

[1] Also referred as POSIX time, is the number of seconds that have elapsed since 00:00:00 Thursday, 1 January 1970

```
ISS (ZARYA)
1 25544U 98067A   08264.51782528 -.00002182  00000-0 -11606-4 0  2927
2 25544  51.6416 247.4627 0006703 130.5360 325.0288 15.72125391563537
```

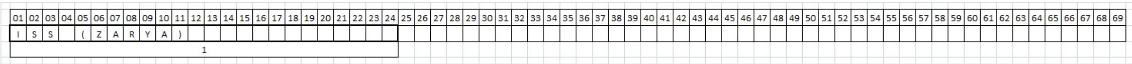


Figure 4.1: Satellite Name

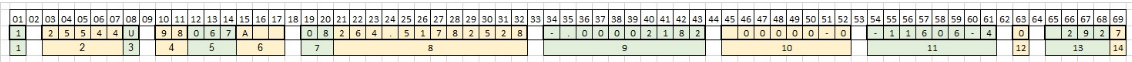


Figure 4.2: First TLE line

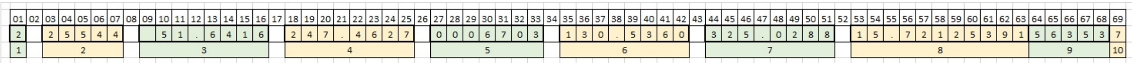


Figure 4.3: Second TLE line
[33]

Table 4.1: TLE parameters description

Line 0			
Field	**Column**	**Description**	**Example**
1	01-24	Satellite name and designation	ISS (ZARYA)
Line1			
Field	**Column**	**Description**	**Example**
1	01	Line Number of Element Data	1
2	03-07	Satellite Number	2544
3	08	Classification (U=Unclassified)	U
4	10-11	International Designator (Last two digits of launch year)	98
5	12-14	International Designator (Launch number of the year)	067
6	15-17	International Designator (Piece of the launch)	A
7	19-20	Epoch Year (Last two digits of year)	08
8	21-32	Epoch (Day of the year and fractional portion of the day)	264.51782528
9	34-43	First Time Derivative of the Mean Motion	-.00002182
10	45-52	Second Time Derivative of Mean Motion (decimal point assumed)	00000-0
11	54-61	BSTAR drag term (decimal point assumed)	-11606-4
12	63	Ephemeris type	0
13	65-68	Element number	292
14	69	Checksum (Modulo 10) (Letters, blanks, periods, plus signs = 0; minus signs = 1)	7
Line 2			
Field	**Column**	**Description**	**Example**
1	01	Line Number of Element Data	2
2	03-07	Satellite Number	25544
3	09-16	Inclination [Degrees]	51.6416
4	18-25	Right Ascension of the Ascending Node [Degrees]	247.4627
5	27-33	Eccentricity (decimal point assumed)	0006703
6	35-42	Argument of Perigee [Degrees]	130.5360
7	44-51	Mean Anomaly [Degrees]	325.0288
8	53-63	Mean Motion [Revs per day]	15.72125391
9	64-68	Revolution number at epoch [Revs]	56353
10	69	Checksum (Modulo 10)	7

[34]

Feeding the algorithm with the data highlighted in yellow in table 4.1 is mandatory in order to let the code perform as expected. In reference to satellite data, two sets of TLE are needed to run the analysis.

4.2 Core algorithm

The first part developed must be the one that solves the visibility at a given time, for only two satellites, and without perturbations because a stationary orbit is considered. Then, through the data extracted from both TLE, the orbit for each satellite is determined for any time. The true anomaly is the parameter that changes, that is the reason why it is calculated inside the algorithm and TLE is not giving that parameter.

The most comfortable way to handle this data is to store it in a *char* structure. Later on, the process of parsing all the data into each parameter will be described.

Hereunder, it is presented, step by step, the basic algorithm as defined in section 3.1.3. It has been simplified. It does not include preallocations, and already known variables definition. After this, only conceptual descriptions will be made to explain the evolution from the core.

```
for i=1:2

a(i) = (mu/n(i)^2 )^(1/3);
q(i) = a(i)*(1-e(i));

% Step 1 - Finding unperturbed mean motion
if e(i) ≥ 0
n(i) = n(i);
else
error('Eccentricity cannot be a negative value')
end

% Step 2 - Solving Mean Anomaly
M(i) = n(i)*(t-T(i));

% Step 3 - Finding true anomaly
if e(i) > 1
Fn(i) = 6*M(i);
error = 1;
while error > 1e-8
    Fn1(i) = Fn(i)+(M(i)-e(i)*sinh(Fn(i))+Fn(i))/(e(i)*cosh(Fn(i))-1);
    error = abs(Fn1(i)-Fn(i));
    Fn(i) = Fn1(i);
end

f(i) = atan((-sinh(Fn(i))*sqrt(e(i)^2-1))/(cosh(Fn(i))-e(i)));

elseif e(i) == 1
A(i) = (3/2)*M(i);
B(i) = (sqrt(A(i)^2+1)+A(i))^(1/3);
C(i) = B(i)-1/B(i);
f(i) = 2*atan(C(i));

elseif e(i) < 1 && e(i) ≥ 0
% Convert mean anomaly to true anomaly.
% First, compute the eccentric anomaly.
Ea = Keplers_Eqn(M(i),e(i));

```

```
% Compute the true anomaly f.
y = sin(Ea)*sqrt(1-e(i)^2)/(1-e(i)*cos(Ea));
z = (cos(Ea)-e(i))/(1-e(i)*cos(Ea));

f(i) = atan2(y,z);

else
error('Eccentricity cannot be a negative value')
end

% Step 4 - Finding primary body center to satellite distance
r(i) = (1+e(i))*q(i)/(1+e(i)*cos(f(i)));

% Step 5 - Finding standard orientation vectors
Px(i) = cos(omega(i))*cos(RAAN(i))-sin(omega(i))*sin(RAAN(i))*cos(inc(i));
Py(i) = cos(omega(i))*sin(RAAN(i))+sin(omega(i))*cos(RAAN(i))*cos(inc(i));
Pz(i) = sin(omega(i))*sin(inc(i));
Qx(i) = -sin(omega(i))*cos(RAAN(i))+cos(omega(i))*sin(RAAN(i))*cos(inc(i));
Qy(i) = -sin(omega(i))*sin(RAAN(i))+cos(omega(i))*cos(RAAN(i))*cos(inc(i));
Qz(i) = cos(omega(i))*sin(inc(i));

% Step 6 - Finding components of the primary body center to satellite ...
    vector in the orbital plane
xi(i) = r(i)*cos(f(i));
eta(i) = r(i)*sin(f(i));

% Step 7 - Finding primary body center to satellite vector
r_fullvector = xi(i)*[Px(i) Py(i) Pz(i)] + eta(i)*[Qx(i) Qy(i) Qz(i)];
for j=1:3
r_vector(i,j) = r_fullvector(j);
end

% Step 8 - Finding Parameter or Semi-parameter
parameter(i) = a(i)*(1-e(i)^2);

end

% Step 9 - Solving visibility equation

sat1=1;
sat2=2;

P1 = [Px(sat1) Py(sat1) Pz(sat1)];
P2 = [Px(sat2) Py(sat2) Pz(sat2)];
Q1 = [Qx(sat1) Qy(sat1) Qz(sat1)];
Q2 = [Qx(sat2) Qy(sat2) Qz(sat2)];
A1 = dot(P1,P2);
A2 = dot(Q1,P2);
A3 = dot(P1,Q2);
A4 = dot(Q1,Q2);

D1 = sqrt(A1^2+A2^2);
D2 = sqrt(A3^2+A4^2);

r1dotr2complex = (parameter(sat1)*parameter(sat2)/...
((1+e(sat1)*cos(f(sat1)))*(1+e(sat2)*...
cos(f(sat2)))))*(D1*cos(f(sat2))*(cos_gamma*cos(f(sat1))+sin_gamma*...
sin(f(sat1)))+D2*sin(f(sat2))*(cos_psi*cos(f(sat1))+sin_psi*sin(f(sat1))));

Rcomplex = parameter(sat1)^2*parameter(sat2)^2*...
(D1*cos(f(sat2))*(cos_gamma*cos(f(sat1))+sin_gamma*sin(f(sat1)))+D2*...
sin(f(sat2))*(cos_psi*cos(f(sat1))+sin_psi*sin(f(sat1))) )^2-...
parameter(sat1)^2*parameter(sat2)^2+S^2*(parameter(sat1)^2*...
```

```
(1+e(sat2)*cos(f(sat2)))^2+parameter(sat2)^2*...
(1+e(sat1)*cos(f(sat1)))^2)-2*S^2*parameter(sat1)...
*parameter(sat2)*(D1*cos(f(sat2))*(cos_gamma*cos(f(sat1))+sin_gamma*...
sin(f(sat1)))+D2*sin(f(sat2))*(cos_psi*cos(f(sat1))+sin_psi*...
sin(f(sat1))))*(1+e(sat1)*cos(f(sat1)))*...
(1+e(sat2)*cos(f(sat2)));

Rv = sqrt((r(sat1)^2 * r(sat2)^2 - r1dotr2complex^2)/(r(sat1)^2...
+ r(sat2)^2 - 2*r1dotr2complex)) - body_radius;

% Step 10: Print Results for the given epoch time
pair_result = 'The result for %s%s and %s%s at %s is %d ';
visibility = '--- Direct line of sight';
non_visibility= '--- Non-visibility';

t_todatetime = datetime(t, 'ConvertFrom', 'posixtime');

result_to_log = sprintf(pair_result, ID{sat1}, designation{sat1}, ...
    ID{sat2}, designation{sat2}, t_todatetime, Rcomplex num);
fprintf(result_to_log); % Command window print

if Rcomplex < 0
disp(visibility); % Command window print
fprintf(fid_log, '%s: %s%s\n', datestr(datetime('now', 'TimeZone', ...
    'UTC')), result_to_log, visibility); % Appending visibility analysis ...
    result to log file
else
disp(non_visibility); % Command window print
fprintf(fid_log, '%s: %s%s\n', datestr(datetime('now', 'TimeZone', ...
    'UTC')), result_to_log, non_visibility); % Appending visibility ...
    analysis result to log file
end
```

4.3 Orbit tracking and simulation time

Once having the basic solution for the visibility between two satellites, the next logic step is two implement a period analysis to find communication windows, not only moments. Then some new parameters need to be defined:

- Simulation start time
- Simulation end time
- Time divisions or increment

This can be achieved by a loop and the determination of the orbit. As discussed previously in section 2.4, there are two kinds of approaches, one being a lot more complex than the other. First one is to consider the stationary orbit. In this case, only the loop in time is needed as the only orbital parameter changing is the true anomaly (f) and this is solved in the previous method. The second one uses a perturbation model and, in contrast, implies almost all used orbital parameters changing over time.

Stationary orbit approach is as follows:

```
for t=t:increment:t_end % Simulation time and time discretization
    for i=1:2
        % CORE ALGORITHM
    end
end
```

For the second approach the code has been adapted to work with the SGP4 function developed by Meysam Mahooti for Matlab in 2017. [35] Perturbation model (SGP4) approach is as follows:

```
for t=t:increment:t_end % Simulation time and time discretization
    for i=1:2
        [pos, vel, OrbitDataProp] = sgp4(tsince, OrbitData, i); % ...
            calling SGP4 function
        % CORE ALGORITHM
    end
end
```

4.4 Constellation module

Satellites' data is stored in structures. This helps in the identification task. If 10 satellites are given as input, in ascending order, we will have satellite 1,2,3,...,8,9,10. With the process designed down below, it is possible to cover all pairs of satellites in an efficient way. Satellite 1 is paired with all the others, satellite 2 with all the others except 1, satellite 3 with all the others except satellite 1 and satellite 2, etc. Using this technique repetitions are avoided.

Notice that the 'for' to analyze a selected pair is also different. Now, it is not only i=1,2 because more satellites with identification numbers greater that 2 appear. The same efficient method to avoid pair repetitions selects the identification for each satellite and through a two-loop 'for' the trick is done by substituting the iterated parameter for the second satellite identifier.

```
OrbitDataProp = OrbitData;
num_pairs = 0;
for sat1=1:num_satellites-1
    for sat2=sat1+1:num_satellites
        num_pairs = num_pairs + 1;
        for t=t:increment:t_end % Simulation time and time discretization
            i = sat1;
            for x=1:2
                [pos, vel, OrbitDataProp] = sgp4(tsince, OrbitData, i);
                % CORE ALGORITHM
                i = sat2;
            end
        end
    end
end
```

4.5 Features

The following subsections are a summary of the main extra features implemented in the code.

4.5.1 Satellites' input

When running a satellite to satellite visibility analysis, there are three different ways to input constellations:

- Hard-coded TLE examples
- TXT TLE file
- Pasting TLE

This module adapts the TLE function developed by Tyler G. R. Reid for Matlab in 2017. [36] Basically, it applies a logic that parses all the given data to build a structure with all the parameters for each satellite. It was a personal development to implement the paste method as it can potentially be a very user-friendly method. The original implementation, also adapted to the study, is a TXT file reading method.

The following figure is presented to understand more the data structure obtained in the code.

```
TLE data collected:
         ID: {'EGYPTSAT'  'TRMM'  'GOES'  'NOAA'  'NAVSTAR'}
designation: {'1'  ''  '3'  '3'  '46'}
        PRN: {''  ''  ''  ''  ''}
          i: [1.711340710140992 0.610285788886353 0.248122987780522 1.776019121561398 0.891252382530904]
       RAAN: [3.818148594174371 0.935260859619941 0.055794685527755 3.000971475756610 3.891105102236986]
      omega: [1.068174663476318 4.732332857747229 5.872785096207135 3.269630007516097 0.573558825436636]
          M: [5.218349458562334 1.551992149433910 0.409483913115154 3.015255250354932 5.719370654956834]
          n: [0.001068932468711 0.001131464427384 7.292569354745049e-05 9.019638721676751e-04 1.458572397656593e-04]
          a: [7.039563745514021e+06 6.777465127946460e+06 4.216242158765066e+07 7.883513766108483e+06 2.655995071588150e+07]
          e: [7.144000000000000e-04 1.034000000000000e-04 1.795000000000000e-04 6.223000000000000e-04 0.007904400000000]
       date: {'21-May-2008 17:49:57.228'  '20-May-2008 20:12:15.399'  '19-May-2008 15:23:30.338'  '20-May-2008 22:13:29.783'  '21-May-2008 03:23:22.576'}
      Bstar: [1.365400000000000e-05 4.191900000000000e-05 1.000000000000000e-04 1.000000000000000e-04 1.000000000000000e-04]
         BC: [5.747982254057621e+03 1.872252431997490e+03 7.848294969690277e+02 7.848294969690277e+02 7.848294969690277e+02]
      epoch: [1.211392197228000e+09 1.211314335399000e+09 1.211210610338000e+09 1.211321609783000e+09 1.211340202576000e+09]
          T: [1.211387315395320e+09 1.211312963732236e+09 1.211204995253752e+09 1.211318266794058e+09 1.211300990462401e+09]
          q: [7.034534681174226e+06 6.777045693805222e+06 4.215485343297567e+07 7.878607855491834e+06 2.635001024144289e+07]
```

Figure 4.4: TLE Data Structure

4.5.2 Celestial object system input

The Celestial object system module lets the user select another primary body different than Earth assuming that TLE for other planets such as Mars are available. The input parameters are the following:

- Primary body radius (a_e)
- Extra distance (Δ)
- Standard gravitational parameter (μ)

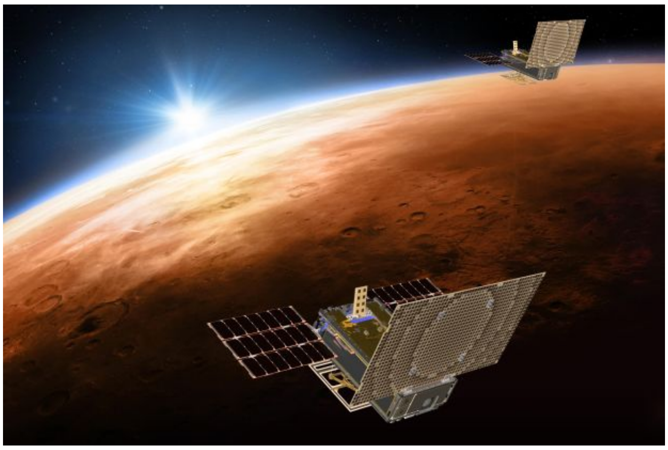


Figure 4.5: *MarCO* satellite heading to Mars
[37]

4.5.3 Simulation time input

In this module the user is requested to run a simulation from now to tomorrow, or to select a custom input following the format given in this example: 22-Jan-2019 13:22:22. It also requires to input the desired time divisions.

4.5.4 Plots and animations

This module runs at the end of the analysis to avoid affecting the run-time of the simulation. In order to do that all, the calculations are stored into preallocated matrix during the analysis.

This module also uses a function by the already previously mentioned Tyler G. R. Reid to adapt to Earth Centered Inertial (ECI) frame and gives that to a 3D Earth Plot developed by Bruno Luong. [38] Again, this was adapted and further developed to the study. The live plot explained below is a personal implementation added for the purpose of the study.

It asks to select one of the following options:

- **Static plot**: displays a 3D plot of the Earth (showing its initial position referring to rotation) and prints the trace each satellite left and its starting point with a unique color for each satellite.
- **Live plot**: same as mentioned above but the plot is printed in real-time step by step (live simulation time). It computes the analysis for each pair of satellite and shows in green if there is direct line of sight between satellites. It also gives as output an MP4 file with the recorded animation.

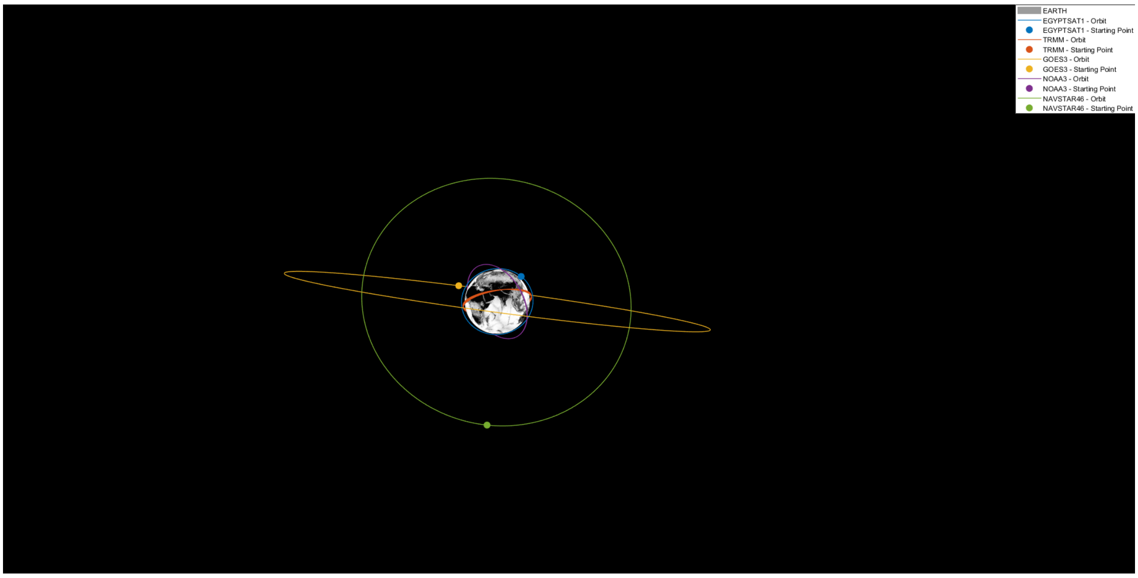


Figure 4.6: Static plot

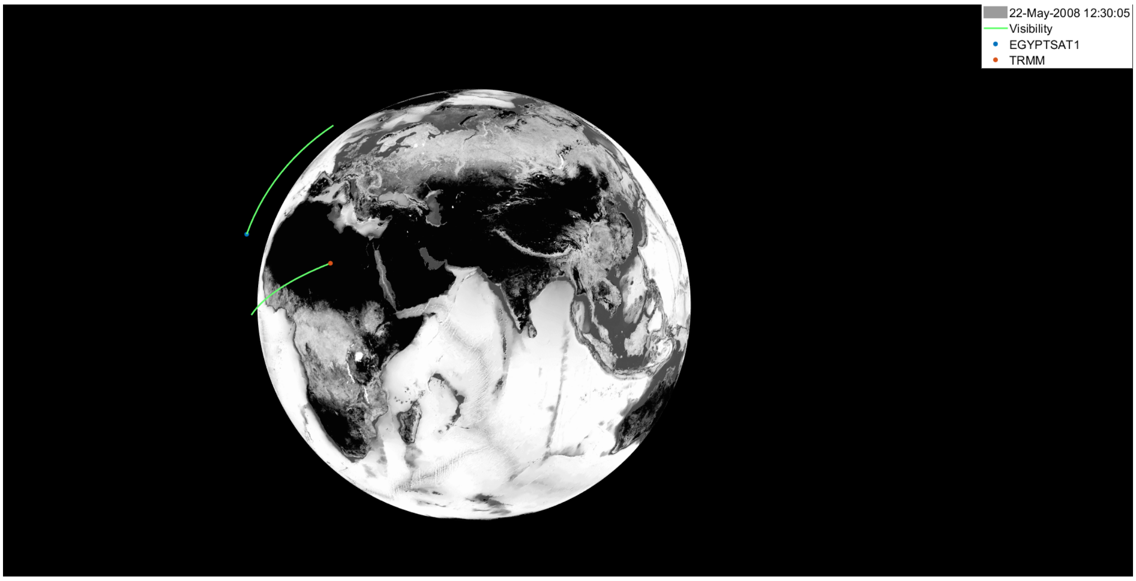


Figure 4.7: Live plot

4.5.5 Logging

In order to register what happens in Matlab while running simulations a LOG file is generated by the code. This lets users to explore simulations - useful in Linux based systems - and work in an agile way when it comes to evaluate and understand the outcome given by the program, even after closing Matlab session.

4.5.6 Data analytics

One of the areas that is experiencing more growth during the last years is data analytics. Following that trend, a structured data model was created to store all the calculations for each pair of satellite and each time step. Then all the data is dumped into a CSV file at the end of the simulation, again, to avoid affecting the run-time of the visibility analysis. Using that information in a data analytics tool - such as *Power BI* - brings the opportunity to quickly navigate through the data and provides a clear understanding. As a consequence, that turns this feature into a very powerful tool.

4.5.7 Pathfinding algorithm

This algorithm is further developed in the next chapter. The basic idea behind is to find the quickest path - in terms of time - between two satellites. It closes the loop for one of the possible final applications of this study. Although it is out of scope, it was worth some research.

5. Communications architecture

Since the early days of the space race, the idea of being capable of communicating using satellites was carefully considered. Even before the space race's start, the potential application of rocketry awoke brilliant ideas in brilliant people such as Arthur C. Clarke. The famous British science fiction writer made accurate predictions years before satellites coming to reality. In his paper *Extra-Terrestrial Relays* published in *Wireless World* in 1945, he stated that contributions in rocketry from Tsiolkovski, or Wernher von Braun - who designed the V2 rockets used in the war that had just ended that same year - could be effectively applied in peaceful pioneer applications.

Many of the first satellites developed at the early stage were based on passive behavior. A clear example of that is *Echo 1*, first NASA's communications satellite, that was basically a balloon with the purpose of mirroring radio communications.

Figure 5.1: *Echo 1* balloon satellite
[39]

Communications architecture refers to the configuration of satellites and ground stations in a space system, and the network of communication links that transfers information between them. The communication links can be either between ground station and satellite or between two satellites. Satellite to satellite communications is sometimes referred to as an Inter-satellite Link (ISL). Communication from ground station to satellite is called uplink while communication from satellite to ground is called downlink.

This architecture is defined by the application and its specifications. As already mentioned in this study in constellations design, this architectures can be defined by:

- **Orbit height**: LEO, MEO, GEO
- **Specific orbit configurations**: Molniya orbits give more coverage in northern latitudes as the perigee is fixed on the southern part of the Earth and the eccentricity leads to spending most of the time covering north.

- **Use of satellite interlink**: making use of the constellation ISL redefines the architecture more than any other parameter. Previous specifications define coverage, and in fact, is the end or start point for a communication. However, interlink between satellites opens a complex and innovative way to transfer data in constellations.
- **RF spectrum**: electromagnetic spectrum of the wave carrying the data. The higher frequency bands typically give access to wider bandwidths, but are also more susceptible to signal degradation due to 'rain fade'[2].
- **Data rate**: the quantity of information per unit time transferred between the satellite and ground station. The higher the data rate, the larger the transmitter power and antenna size required.
- **Link availability**: visibility windows between ground stations or users, and satellites; and between satellites in constellations.

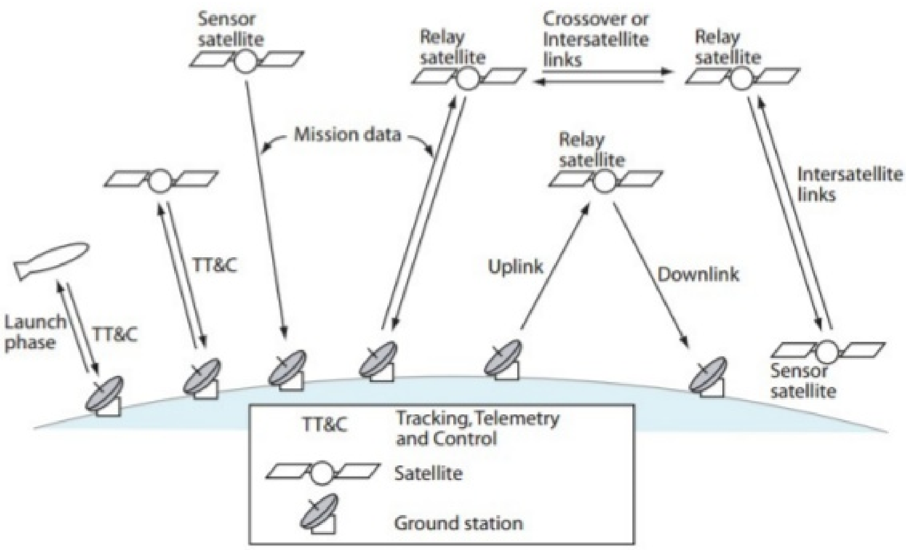


Figure 5.2: Basic satellite communication links
[40]

As far as the communication bands are concerned, the following are the commonly used:

- Very High Frequency (VHF): 30 to 300 MHz
- Ultra High Frequency (UHF): 300 MHz to 3 GHz
- L band: 1 to 2 GHz
- S band: 2 to 4 GHz
- C band: 4 to 8 GHz
- X band: 8 to 12 GHz
- Ku band: 12 to 18 GHz

[2]The absorption of radio signals by atmospheric rain, snow or ice

- K band: 18 to 27 GHz
- Ka band: 27 to 40 GHz
- Optical (Laser Communication): 100 to 800 THz

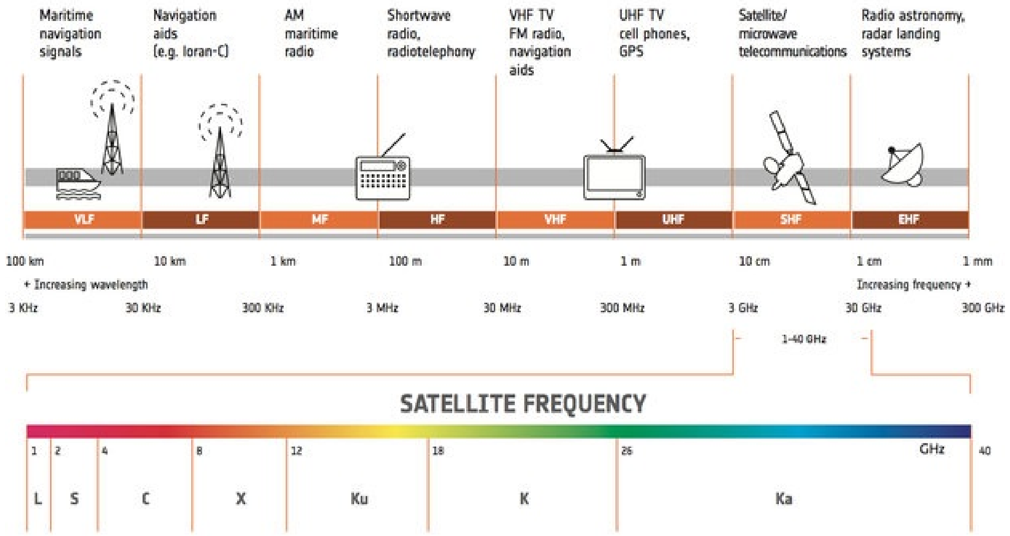


Figure 5.3: Frequency bands involved in communications
[41]

Focusing on data manipulations, the basic steps are encoding/decoding, modulation/demodulation and forward error correction. Encoding refers to changing the format of the data or encrypting. Modulation is about changing the frequency and the amplitude of the carrier. Lastly, forward error correction designates the process to control errors in data transmission due to noise introduced in the signal by the environment.

Without any doubt, communication is one of the most important application for satellites. This present study brings a solution to the visibility windows problem between satellites. That is only an intermediate step. The final utilization for the results obtained is a better planning for data transfers in current constellations making use of the visibility paths discovered. Additionally, it can be a useful tool to design new constellations based on an enhanced efficient communication.

5.1 Data transfer time optimization

The ability to quickly share data between satellites leads to new possibilities in the industry. *CubeSats* are becoming a promising platform. They are significantly more affordable to build and launch but, at the same time, they struggle to transmit high-rate data down to Earth due to power and size limitations. This is improving thanks to advances in previously mentioned laser communications. [42]

"The new laser-pointing platform for *CubeSats*, which is detailed in the journal Optical Engineering, enables CubeSats to downlink data using fewer on-board resources at significantly higher rates than is currently possible. Rather than send down only a few images each time a

CubeSat passes over a ground station, the satellites should be able to downlink thousands of high-resolution images with each flyby." [43]

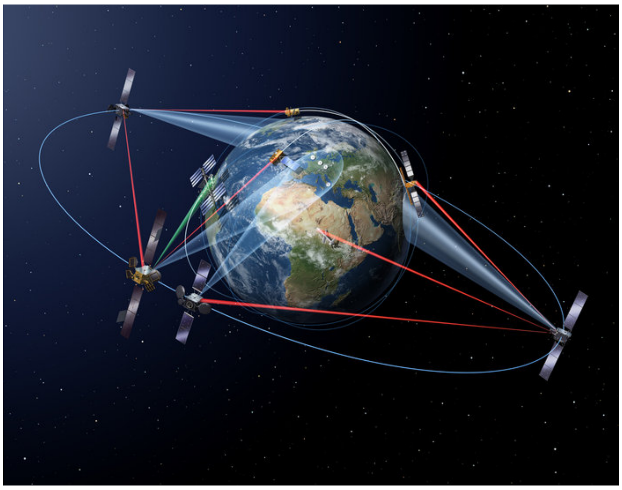


Figure 5.4: Laser inter-satellite data link
[44]

Previous quote describing the advances in laser communication for *CubeSats* - nonetheless, can be applied to satellites in general - leave an important statement. It claims that images are sent every time a satellite passes through ground stations. Applying the same optic link principles between satellites and managing the network provided by the constellation can be a game-changer. It is setting a new definition for real-time data. The key advantages are the following:

- **Low latency**: if it is efficiently implemented, sending data between satellites in a LEO constellation should not add much latency because distances are still short compared to MEO and GEO satellites.
- **Full indirect coverage**: real-time could be ideally true. Things happening in one site of the Earth could reach a ground station on the other site thanks to the satellite network provided by the constellation.
- **Load balanced**: to achieve an efficient work load in each satellite, the paths should be optimized. Instead of sending the data to all visible satellites, the quickest path can be determined beforehand using the solution provided by the algorithm explained hereafter.

As always, it comes with some downsides. The laser is affected by atmospheric conditions. Humidity can degrade speeds while fog and clouds can even block transmissions. However, the presented algorithm is not bond to any specific technology. It is only a pathfinding algorithm that can serve all technologies involved in satellite communication systems.

5.2 Pathfinding algorithm

Finding the shortest path between two satellites is a quite complex problem. Take into account a large satellite constellation and think about the amount of possible paths and the way to explore them.

This problem has a long story in graph structured data. It is called pathfinding and it is applied in *Google Maps,* for instance. Locations are represented as nodes and, distance between locations, as arrows with a distance value.

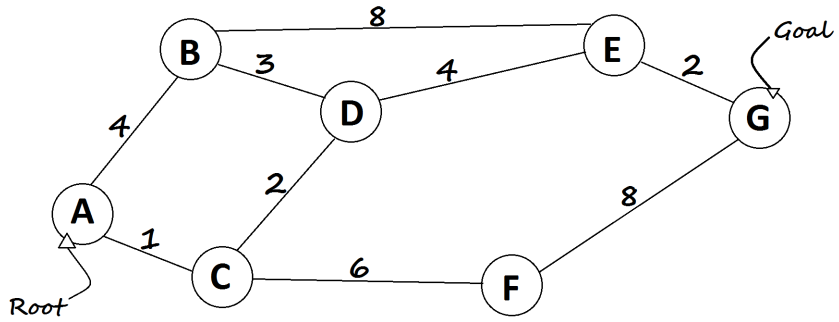


Figure 5.5: Graph structure
[45]

Applying algorithms such as A* or Dijkstra, it is possible to quickly determine the shortest path between two nodes. It is not necessary to examine all possible paths. This kind of algorithms eliminate paths either using educated guesses (heuristics) or distance from source to final node to find the optimal path.

In the case studied here some **key problems** are found:

- All nodes are related to each other.
- Instead of distance, link value is time. The implication of that is huge. Each link value dynamically changes depending on the path evaluated. The visibility window time that links two satellites has to be later than the last evaluated relationship.

Then, to approach the problem some **hypothesis and requirements** will be set:

1. The maximum number of jumps from sender to receiver is 3. It is considered that situations where more jumps than three involve shortest time paths are improbable.
2. Travel time for the data from the sender to the receiver is neglected. However, it can be included as a constant value in the transfer time defined by the user to simulate data processing for the receiver.
3. Going back to an already evaluated satellite is not possible. It would be inefficient as that is not a logic time-saving path.
4. Time can not go backwards at any time.
5. Obviously, the analysis will be made for the period studied in the simulation.

The **custom algorithm** implemented consists in the following basic steps:

1. Input of satellite sender, satellite receiver and transfer time.
2. Creation of a data structure containing the start time, the end time and the total time for all visibility windows for each pair of satellites. It is needed to store them chronologically.

3. Modifying the previous data structure to store and order the visibility windows that last longer than the transfer time given as input.

4. Evaluating the path for 1, 2 and 3 jumps. Sender and receiver are known. The other candidates are all tested adapting the time value to be greater than the previous one, as time must not go backwards at any time.

Down below a sample on how the visibility windows are analyzed per pairs and ordered by the condition of lasting longer than the input transfer data for the two-jump path. The complete code can be found in the *Annexes* document.

```
% Windows per pair able to transfer the required data
for sat1=1:num_satellites-1

for sat2=sat1+1:num_satellites
i=sat1;
j=sat2;
for x=1:2
% 10 Windows per pair
num_windows = 1;
for y=1:10
while WindowsData.time(i,j,num_windows) < transfer_time && num_windows ...
    < length(WindowsData.time)
num_windows = num_windows + 1;
end
if WindowsData.time(i,j,num_windows) > transfer_time && ...
    WindowsData.start(i,j,num_windows) > 0
WindowsDataFirst.start(i,j,y) =  WindowsData.start(i,j,num_windows);
WindowsDataFirst.end(i,j,y) = WindowsData.end(i,j,num_windows);
WindowsDataFirst.time(i,j,y) =  WindowsData.time(i,j,num_windows);
num_windows = num_windows + 1;
end
end

i=sat2;
j=sat1;

end

end

end

% Two Jumps Path
if num_satellites > 2
index_count = 0;
for x=1:num_satellites
y = x;
while y == end_sat || y == start_sat
y = y+1;
end
if y > num_satellites
else
index_count = index_count + 1;
PathSolution2.sat_start(index_count,1) = start_sat;
PathSolution2.sat_end(index_count,1) = y;
PathSolution2.start(index_count,1) = WindowsDataFirst.start(start_sat,y,1);
PathSolution2.end(index_count,1) = ...
    WindowsDataFirst.start(start_sat,y,1) + transfer_time;
PathSolution2.total_time(index_count,1) = ...
    PathSolution2.end(index_count,1) - start_time_unix;
```

```

num_windows = 1;
k = num_windows;
while num_windows ≤ 10 && ...
    WindowsDataFirst.end(y,end_sat,num_windows)-transfer_time < ...
    PathSolution2.end(index_count,1)
num_windows = num_windows + 1;
k = num_windows;
end
if k > 10
else
PathSolution2.sat_start(index_count,2) = y;
PathSolution2.sat_end(index_count,2) = end_sat;
if PathSolution2.end(index_count,1) < WindowsDataFirst.start(y,end_sat,k)
PathSolution2.start(index_count,2) = WindowsDataFirst.start(y,end_sat,k);
else
PathSolution2.start(index_count,2) = PathSolution2.end(index_count,1);
end
PathSolution2.end(index_count,2) = PathSolution2.start(index_count,2) + ...
    transfer_time;
PathSolution2.total_time(index_count,2) = ...
    PathSolution2.end(index_count,2) - start_time_unix;
end
end
end
end
```

6. Results

6.1 Visibility windows

The results obtained using the code will be presented resolving the visibility windows for representative constellations. The selected constellations are described down below:

Table 6.1: Tested constellations

Orbit type	Name	Company	Number of satellites	Description
LEO	Iridium (original)	Motorola	34	Provides L-band voice and data coverage to satellite phones, pagers and integrated transceivers over the entire Earth surface.
MEO	O3B	O3b Networks Ltd.	20	Ka-based satellites providing voice and data communications to mobile operators and Internet service providers.
GEO	Anik	Telesat Canada	4	Not an actual constellation but will be used as a test for geostationary constellations.

The analysis will compare the results achieved taking into account perturbations, using the SGP4 function. The atmosphere effect is set to 20,000 meters. All TLE data - for all satellites in the previously mentioned constellations - is provided by NORAD and can be found in *Celestrak*, web run by Dr. T.S. Kelso. [46]

The following results are analyzed with the help of *Power BI*, fed with all the data extracted thanks to the data analytics module implemented in the code.

6.1.1 Iridium analysis

This analysis solved the visibility problem for 36 (34 in the constallation and 2 dummy ones) satellites during 24 hours (from 2019-05-07 12:00:00 to 2019-05-08 12:00:00). This means 666 combinations of satellite pairs. The number of time divisions (steps - 1) in the simulation for each pair is 1,000. The whole simulation solved the visibility problem 666,666 times and took 35 minutes.

TLE data was obtained on May 7, 2019 from *Celestrak*. Check the *Annexes* document to see the exact data. The following were the satellites given to the code as input:

- IRIDIUM 7
- IRIDIUM 5
- IRIDIUM 4
- IRIDIUM 914
- IRIDIUM 16
- IRIDIUM 911
- IRIDIUM 17
- IRIDIUM 920
- IRIDIUM 921
- IRIDIUM 26
- IRIDIUM 46
- IRIDIUM 22
- IRIDIUM 29
- IRIDIUM 33
- IRIDIUM 28
- IRIDIUM 36
- IRIDIUM 39
- IRIDIUM 38
- IRIDIUM 42
- IRIDIUM 44
- IRIDIUM 45
- IRIDIUM 24
- IRIDIUM 54
- IRIDIUM 51
- IRIDIUM 61
- IRIDIUM 57
- IRIDIUM 63

- IRIDIUM 69
- IRIDIUM 71
- IRIDIUM 73
- IRIDIUM 82
- IRIDIUM 2
- IRIDIUM 96
- IRIDIUM 97

As expected for the constellation design of this LEO constellation we have many pairs where we have permanent visibility many other with permanent non-visibility. A specific case where something interesting happens is the analysis for IRIDIUM 7 and IRIDIUM 2. Down below, the plots:

1. Visibility periods: treated as a boolean, 1 means visibility 0 means non-visibility.
2. Visibility result: parameter in the algorithm that decides if there is visibility between two satellites. Negative means visibility and positive, non-visibility.
3. Visibility margin: this parameter is the distance from the surface of the Earth to the line traced between satellites.

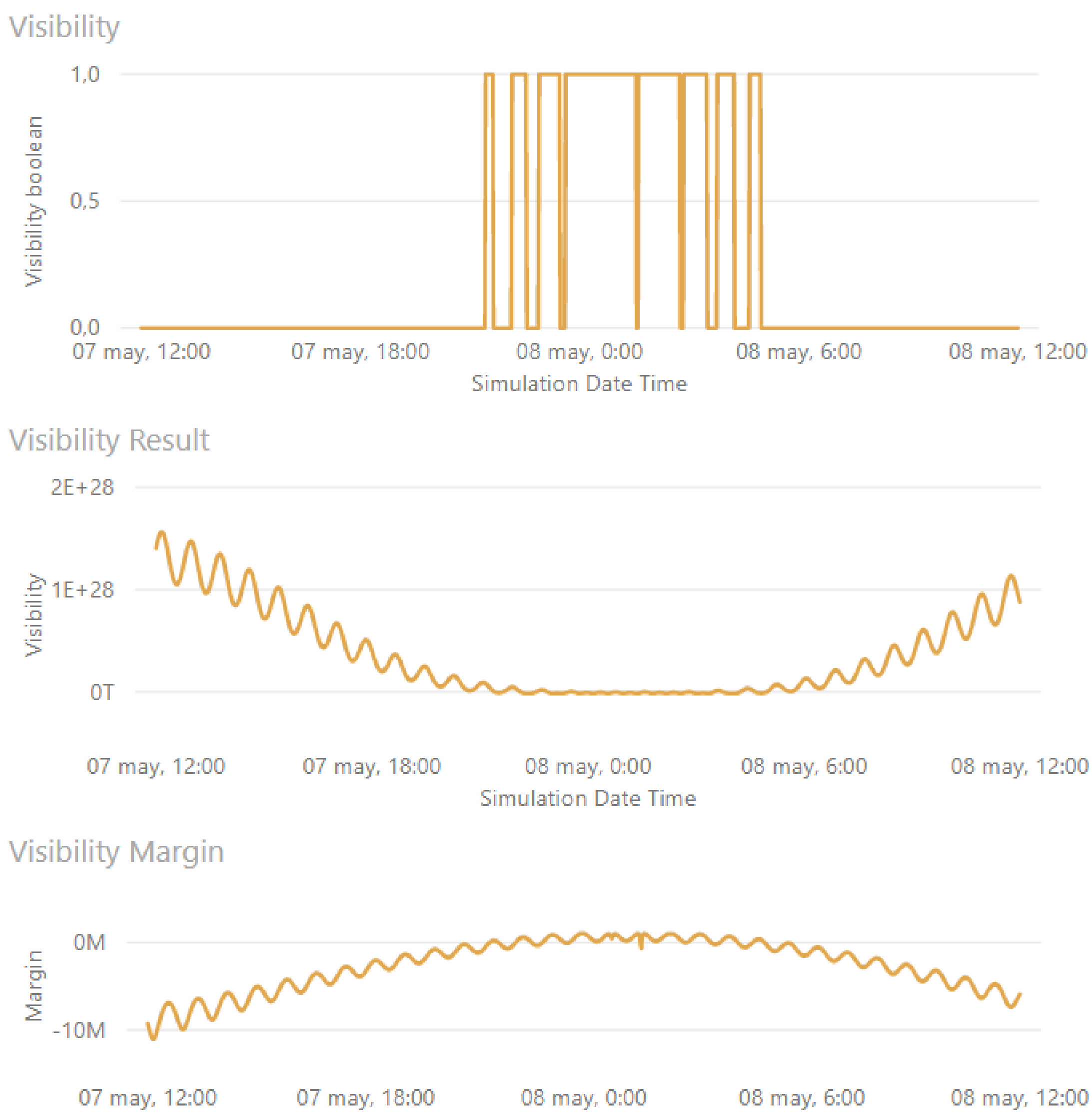


Figure 6.1: IRIDIUM 7 - IRIDIUM 2 visibility analysis

As it can be seen in the plots, the behavior is driven by the periodicity of the orbital movement. The wave form is pretty characteristic. In this case, most of the time visibility is not possible. However, there is a period where visibility is good with only few interruptions.

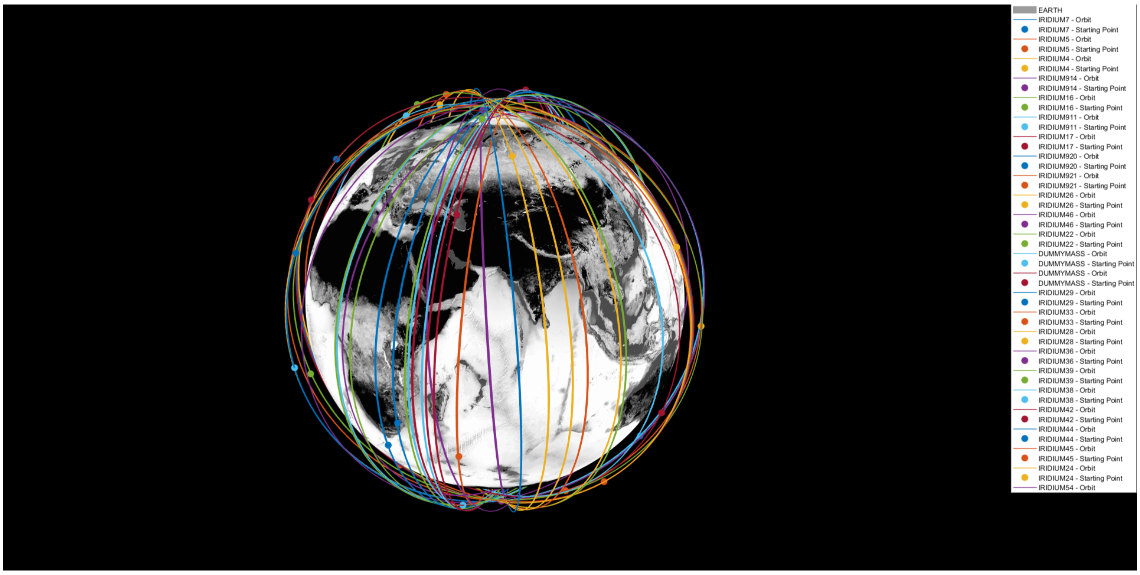


Figure 6.2: Orbit plots for Iridium simulation

6.1.2 O3B analysis

This analysis solved the visibility problem for 20 satellites during 24 hours (from 2019-05-07 12:00:00 to 2019-05-08 12:00:00). This means 210 combinations of satellite pairs. The number of time divisions (steps - 1) in the simulation for each pair is 1,000. The whole simulation solved the visibility problem 210,210 times and took 11 minutes.

TLE data was obtained on May 7, 2019 from *Celestrak*. Check the *Annexes* document to see the exact data. The following were the satellites given to the code as input:

- O3B FM5
- O3B FM4
- O3B FM2
- O3B PFM
- O3B FM3
- O3B FM7
- O3B FM6
- O3B FM8
- O3B FM10
- O3B FM11
- O3B FM12
- O3B FM9
- O3B FM15
- O3B FM16
- O3B FM14
- O3B FM13
- O3B FM20
- O3B FM19
- O3B FM17
- O3B FM18

As expected for the constellation design of this MEO constellation we have more general visibility compared to previously analyzed LEO constellation due to the greater orbit heights. The chosen is the analysis for O3B FM10 and O3B FM16. Down below, the plots:

1. Visibility periods: treated as a boolean, 1 means visibility 0 means non-visibility.

2. Visibility result: parameter in the algorithm that decides if there is visibility between two satellites. Negative means visibility and positive, non-visibility.

3. Visibility margin: this parameter is the distance from the surface of the Earth to the line traced between satellites.

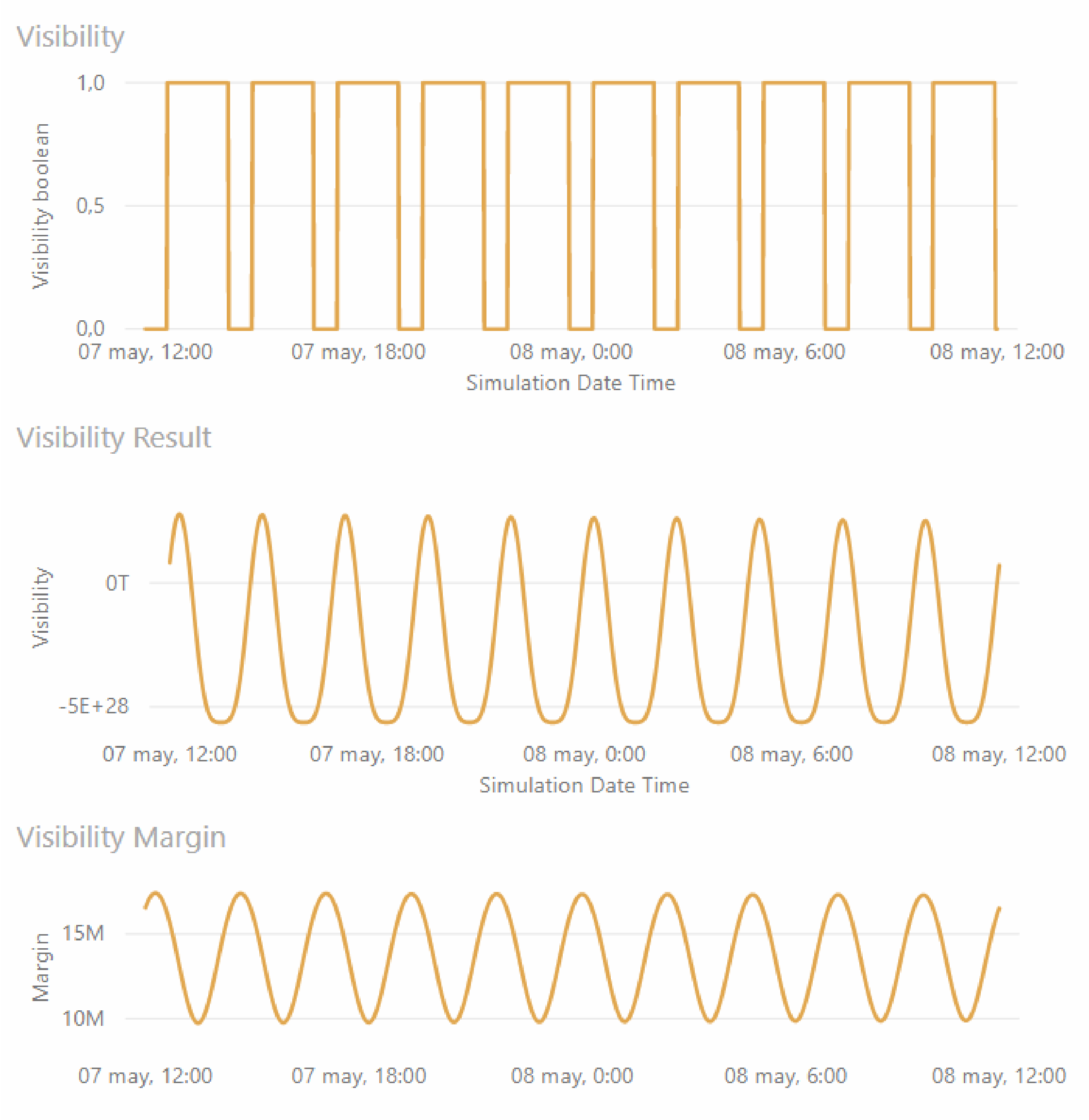


Figure 6.3: O3B FM10 - O3B FM16 visibility analysis

As it can be seen in the plots, the behavior is driven, again, by the periodicity of the orbital movement. In this case, visibility is a lot more periodic between these two satellites. The distance leads to quicker changes in visibility periods. However, this particular case seems that should have permanent visibility or permanent non-visibility due to the orbit being so similar.

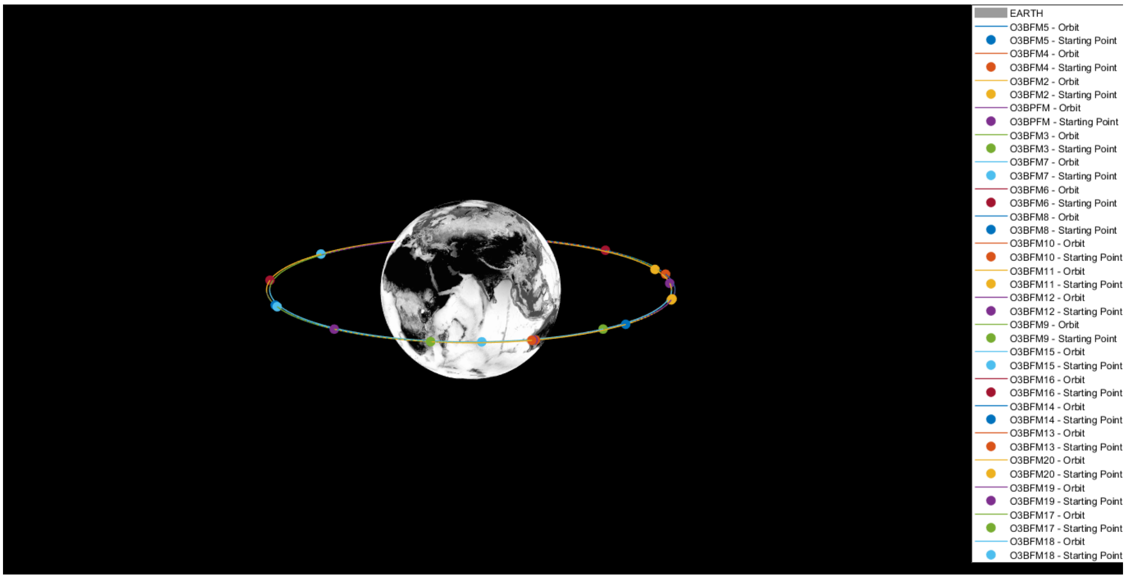


Figure 6.4: Orbit plots for O3B simulation

6.1.3 Anik satellites analysis

This analysis solved the visibility problem for 5 satellites during 24 hours (from 2019-05-07 12:00:00 to 2019-05-08 12:00:00). This means 15 combinations of satellite pairs. The number of time divisions (steps - 1) in the simulation for each pair is 1,000. The whole simulation solved the visibility problem 15,015 times and took 35 seconds.

TLE data was obtained on May 7, 2019 from *Celestrak*. Check the *Annexes* document to see the exact data. The following were the satellites given to the code as input:

- ANIK G1
- ANIK F2
- ANIK F1R
- ANIK F3
- ANIK G1

As expected for the constellation design of this MEO constellation we have more general visibility compared to previously analyzed LEO constellation due to the greater orbit heights. Unless the geometry places two satellites on opposite sites of the Earth, there is permanent visibility between satellites. The chosen is the analysis for ANIK F1R and ANIK G1. Down below, the plots:

1. Visibility periods: treated as a boolean, 1 means visibility 0 means non-visibility.
2. Visibility result: parameter in the algorithm that decides if there is visibility between two satellites. Negative means visibility and positive, non-visibility.
3. Visibility margin: this parameter is the distance from the surface of the Earth to the line traced between satellites.

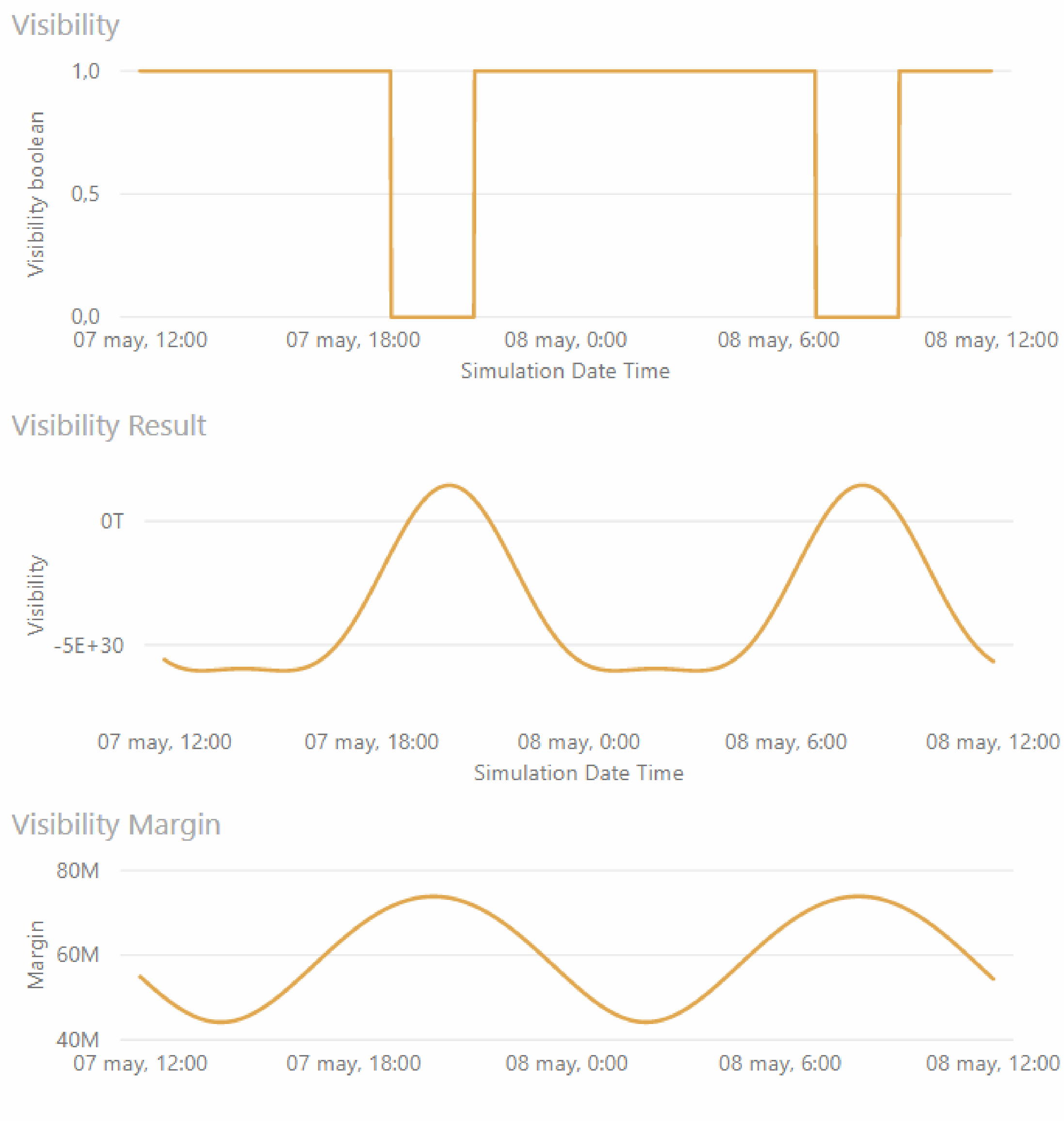


Figure 6.5: Anik F1R - Anik G1 visibility analysis

As it can be seen in the plots, the behavior is driven, again, by the periodicity of the orbital movement. In this case, visibility is like in the previous MEO case but with less non-visibility periods because of the GEO distance. However, this particular case, again, should have permanent visibility or permanent non-visibility due to the orbit being so similar.

In figure 6.6, plot shows three starting points for four satellites. One satellite starting point is that close to other satellite starting point that it can not be shown properly.

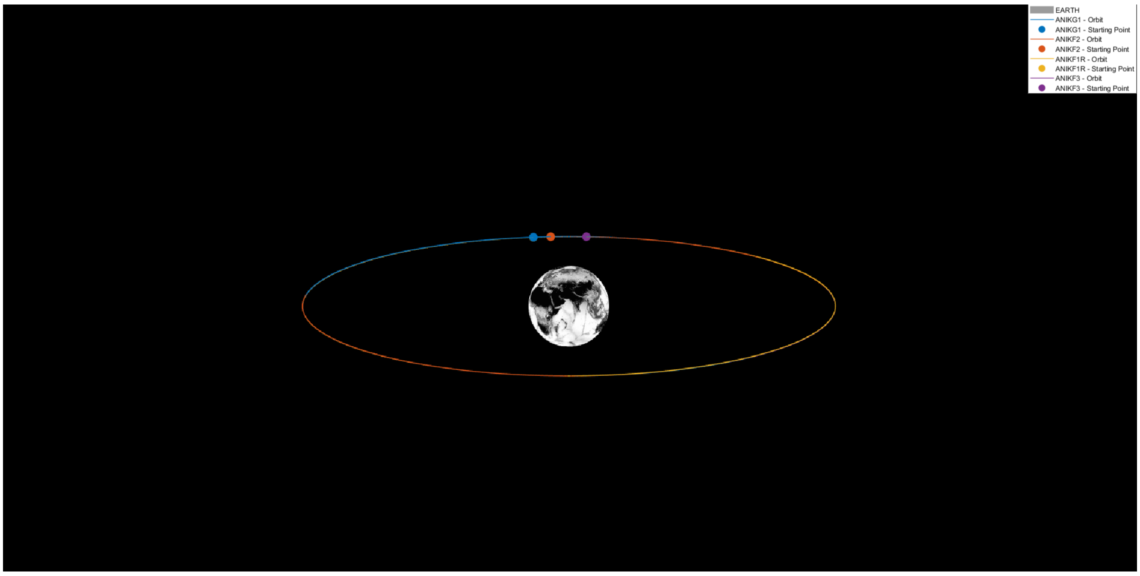


Figure 6.6: Orbit plots for Anik satellites simulation

6.2 Pathfinding

Following the approach set in the previous section, some inter-satellite data transfers will be evaluated for each representative constellation. As implemented in the code, three cases are going to be simulated:

- **One jump**: direct contact between the two subject satellites, sender and receiver.
- **Two jumps**: forcing to send the data to an intermediate satellite before reaching the actual receiver.
- **Three jumps**: forcing to send the data to two intermediate satellites before reaching the actual receiver.

Transfer data time is set to 100 seconds and for the same constellations and time studied in the previous section (from 2019-05-07 12:00:00 to 2019-05-08 12:00:00).

6.2.1 Pathfinder in LEO

Table 6.2: One-jump LEO case

One-jump case	**Sender**	**Receiver**
Satellite	IRIDIUM 7	IRIDIUM 2
Start timestamp	07/05/2019 21:25:55	
End timestamp	07/05/2019 21:27:35	
Time since simulation start	34055.2 seconds	

Table 6.3: Two-jumps LEO case

Two -jumps case	Sender	Intermediate	Intermediate	Receiver
Jump	Jump 1		Jump2	
Satellite	IRIDIUM 7	IRIDIUM4	IRIDIUM4	IRIDIUM 2
Start timestamp	07/05/2019 12:00:00		07/05/2019 19:00:28	
End timestamp	07/05/2019 12:01:40		07/05/2019 19:02:08	
Total since simulation start	100 seconds		25328.8 seconds	

Table 6.4: Three-jumps LEO case

Three -jumps case	Sender	Intermediate 1	Intermediate 1	Intermediate 2	Intermediate 2	Receiver
Jump	Jump 1		Jump2		Jump 3	
Satellite	IRIDIUM 7	DUMMY MASS 2	DUMMY MASS 2	IRIDIUM 71	IRIDIUM 71	IRIDIUM 2
Start timestamp	07/05/2019 12:00:00		07/05/2019 12:34:33		07/05/2019 12:36:13	
End timestamp	07/05/2019 12:01:40		07/05/2019 12:36:13		07/05/2019 12:37:53	
Total since simulation start	100 seconds		2173.6 seconds		2273.6 seconds	

Taking a look at the tables it is clear that the three jump case is the quickest to go from sender to receiver. The specific sender and receiver were chosen knowing that the first visibility window between them would be late in the simulation, as presented in the previous section. It is pretty interesting to notice that the algorithm is able to find a better path for the three-jumps case than the two-jumps case.

6.2.2 Pathfinder in MEO

Table 6.5: One-jump MEO case

One-jump case	Sender	Receiver
Satellite	O3BFM5	O3BFM9
Start timestamp	07/05/2019 12:00:00	
End timestamp	07/05/2019 12:01:40	
Total since simulation start	100 seconds	

Table 6.6: Two-jumps MEO case

Three -jumps case	Sender	Intermediate 1	Intermediate 1	Intermediate 2
Jump	Jump 1		Jump2	
Satellite	O3BFM5	O3BFM4	O3BFM4	O3BFM9
Start timestamp	07/05/2019 12:00:00		07/05/2019 12:01:40	
End timestamp	07/05/2019 12:01:40		07/05/2019 12:03:20	
Total since simulation start	100 seconds		200 seconds	

Table 6.7: Three-jumps MEO case

Three -jumps case	Sender	Intermediate 1	Intermediate 1	Intermediate 2	Intermediate 2	Receiver
Jump	Jump 1		Jump2		Jump 3	
Satellite	O3BFM5	O3BFM4	O3BFM4	O3BFM2	O3BFM2	O3BFM9
Start timestamp	07/05/2019 12:00:00		07/05/2019 12:01:40		07/05/2019 12:03:20	
End timestamp	07/05/2019 12:01:40		07/05/2019 12:03:20		07/05/2019 12:05:00	
Total since simulation start	100 seconds		200 seconds		300 seconds	

In MEO analysis pathfinder algorithm takes full advantage of the almost permanent visibility. That is the reason why it can be observed a non-waiting behaviour every time it jumps. However, one jump is enough to be the quickest in this cases.

6.2.3 Pathfinder in GEO

Table 6.8: One-jump GEO case

One-jump case	**Sender**	**Receiver**
Satellite	ANIKG1	ANIKF2
Start timestamp	07/05/2019 12:00:00	
End timestamp	07/05/2019 12:01:40	
Total since simulation start	100 seconds	

Table 6.9: Two-jumps GEO case

Three -jumps case	**Sender**	**Intermediate 1**	**Intermediate 1**	**Intermediate 2**
Jump	**Jump 1**		**Jump2**	
Satellite	ANIKG1	ANIKF1R	ANIKF1R	ANIKF2
Start timestamp	07/05/2019 12:00:00		07/05/2019 12:01:40	
End timestamp	07/05/2019 12:01:40		07/05/2019 12:03:20	
Total since simulation start	100 seconds		200 seconds	

Table 6.10: Three-jumps GEO case

Three -jumps case	**Sender**	**Intermediate 1**	**Intermediate 1**	**Intermediate 2**	**Intermediate 2**	**Receiver**
Jump	**Jump 1**		**Jump2**		**Jump 3**	
Satellite	ANIKG1	ANIKF1R	ANIKF1R	ANIKF3	ANIKF3	ANIKF2
Start timestamp	07/05/2019 12:00:00		07/05/2019 12:01:40		07/05/2019 12:13:11	
End timestamp	07/05/2019 12:01:40		07/05/2019 12:13:11		07/05/2019 12:14:51	
Total since simulation start	100 seconds		791.2 seconds		891.2 seconds	

In GEO analysis pathfinder algorithm takes full advantage of the almost permanent visibility except for the three-jumps case. In that case, as it is forced to jump to two intermediate satellites, it can be observed that it has to send the data to all satellites in the constellation (only 4 satellites in this constellation). There is some waiting in the second jump. This seems to be one of those short periods where the code predicts non-visibility erroneously.

6.3 Validation

The paper in which the suggested algorithm is heavily inspired, provides some data related to some pairs of satellites. [30] The results will be compared using the same TLE data but without knowing the exact parameter set as extra distance to take into account atmosphere effects. It will be set, again, at 20,000 meters. This simulation goes from 2008-05-22 12:00:00 to 2008-05-23 12:00:00.

Taking the analysis of EGYPTSAT 1 and TRMM what can be observed is that the visibility periods are exactly the same if SGP4 is deactivated. When turned on differences are very subtle due to specific situations in which the change in the orbital motion is enough to put satellites out of visibility.

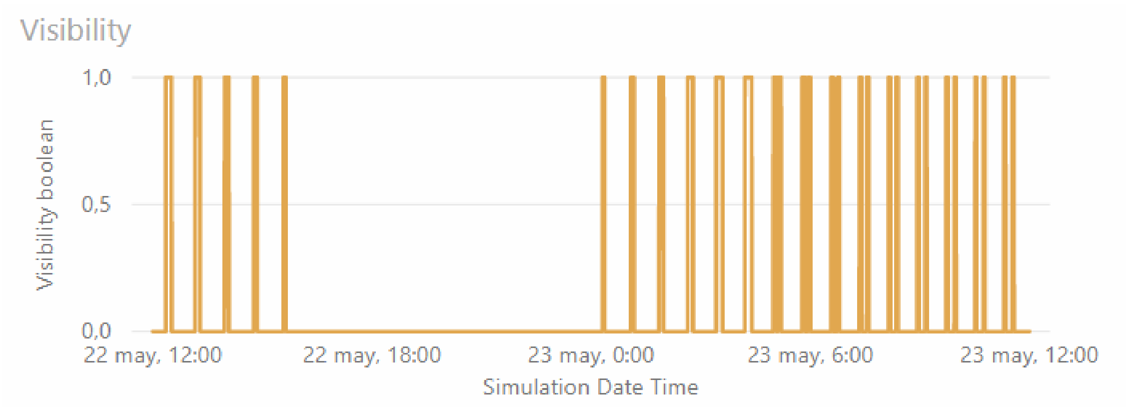


Figure 6.7: EGYPTSAT 1 - TRMM visibility analysis with SGP4

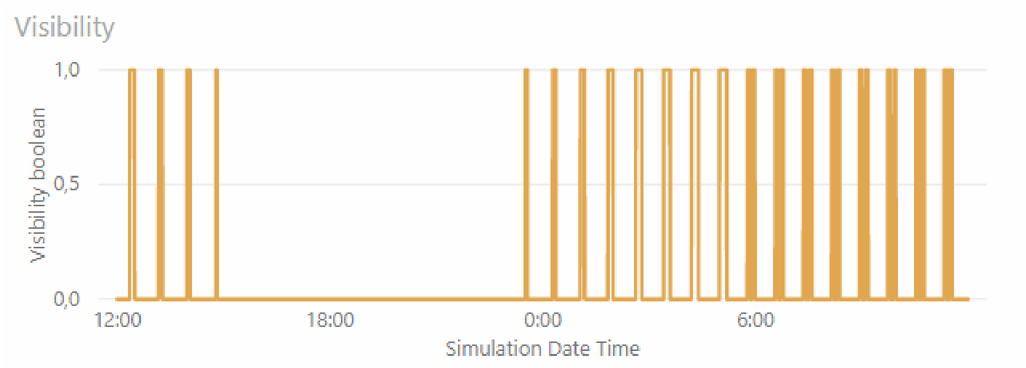


Figure 6.8: EGYPTSAT 1 - TRMM visibility analysis without SGP4

Table 1 EGYPTSAT_1 and TRMM are visible during the times.

Date			Time			to	Date			Time		
Year	Month	Day	Hour	Minute	Seconds		Year	Month	Day	Hour	Minute	Seconds
2008	5	22	12	22	25.99	To	2008	5	22	12	29	50.99
			13	10	39.98					13	16	49.98
			13	58	56.96					14	3	41.96
			14	47	26.95					14	50	24.94
			22	43	18.79					22	46	30.79
			23	30	05.78					23	34	54.77
2008	5	23	0	16	56.77	to	2008	5	23	0	23	15.76
			1	3	58.74					1	11	25.74
			1	50	59.73					1	59	35.73
			2	38	07.71					2	47	37.71
			3	25	15.70					3	35	40.70
			4	12	30.68					4	23	36.68
			4	59	45.67					5	11	31.66
			5	47	06.65					5	59	20.65
			6	34	28.64					6	47	08.63
			7	21	57.62					7	34	50.62
			8	9	26.61					8	15	16.60
			8	16	39.60					8	22	30.60
			8	57	02.59					9	2	52.59
			9	4	16.59					9	10	4.59
			9	44	39.58					9	57	36.57
			10	32	23.56					10	45	03.56
			11	20	08.54					11	32	28.54

Figure 6.9: Visibility periods for EGYPTSAT 1 and TRMM
[30]

To conclude, the code seems to be working properly when satellites are not in the same orbit. In MEO, and GEO cases studied, the code works most of the time but at short periods it predicts non-visibility where visibility is supposed to happen.

7. Conclusions

This thesis described a method for the rapid determination of visibility windows between satellites in a constellation. It is clear that a key step in any space mission is the communication between ground and the space segment. Without this connection very few applications can be conceived. This project studied a more specific relationship inside space missions. The inter-satellite connection. Not all satellites, nor all the constellations make use of inter-satellite communications. However, for those which do, it is highly valuable to have the possibility to predict relative rises and sets between them in order to describe visibility periods. This knowledge leads to the actual use case: ability to plan data transfers.

At the beginning it was difficult and slow, it took many time to see a clear path, to see a promising way to find good results. Nonetheless, with the proper introduction to orbital mechanics and the specific context on how to approach the problem, things started to clarify. A well deserved mention is the fantastic work done by M.A. Sharaf, M.E. Awad, I.A. Hassan, R. Ghoneim and W.N. Ahmed in the paper "Visual contact for two satellites orbits under J2-gravity" in 2012. [30] Their advances in the field made possible the core algorithm created in this thesis and offered a validation framework.

Reviewing the work done, there was so much to be done regarding the problem stated. The limited amount of time led to prioritize some aspects.

First of all, the geometry problem. Being able to solve the geometry in a generic way was something worth a big effort. Lastly, that problem was known to have a feasible solution, giving and elegant result. Depending on the sign of the equation visibility happen or not. Considering the positions of each satellite - in the same reference system - were known for a given time, the geometry itself only gave the chance to compare satellites at a specific moment in time. At this point, it was clear that a tracking algorithm was a must to turn the solution into something truly useful.

So then, it was time to study if all the parameters to deduce the positions required were actually available. That was a reality thanks to the two-line elements (TLE), describing satellites motion in their orbits. This should definitely be a required input. This can be seen as a weak point for the Interlink code, without TLE data code cannot run. Fortunately, it is the standard.

The parameters required are given in the TLE for a specific epoch time. The basic algorithm finds the position for the desired time from a known point and known time provided by the information given by the TLE. At this point the period analysis was available creating the required loops.

After that, a valuable thing to implement was the orbit propagation module. Due to perturbation forces, orbits change over time. Obviously, this affects the position of the satellite, the geometry of the problem and, finally, the visibility result. An SGP4 function was adapted and implemented after some research. This meant a more accurate analysis.

Validation at this point was necessary. As mentioned before, it could be done comparing to the results shown in the paper "Visual contact for two satellites orbits under J2-gravity". The results matched when SGP4 was deactivated, revealing that the calculations made in that paper lacked a perturbations model. However, activating the SGP4 module, the result were almost the same, only having little differences coming from specific cases were visibility was by a very little margin.

The extra features implemented in *Interlink* deserve to be mentioned. Following the agile method set for the development of the code, allowed multiple implementations. As there was not a specific schedule related to the code, it was possible to work in parallel and choose what was useful and possible to implement.

Different TLE input methods, celestial system input choice, simulation time input, logging, meaningful plots and animations were some good extra features. However, the powerful ones are data analytics module and the pathfinding algorithm.

Data analytics provides a game-changing perspective for the results obtained by *Interlink* code. With all the data extracted and structured in a CSV file, it is easy to get custom plots. In this case, the implementation in *Power BI* enables an interactive scrutiny of the data. It is really helpful for, not only results analysis, but also comparing and validating much faster than using Matlab itself.

The pathfinder algorithm is probably the most interesting feature. Although it was not in the scope, the feature is key for a data transfer optimization. Taking advantage of the results obtained, the code finds the best way to send data within a constellation. It has to be considered as a potential use case for real space missions. This is something invisible to the naked eye and can also lead to a better constellation design in constellations that make an extensive use of inter-satellite communications.

It is true that the *Interlink* code is useful as it achieves the goal set at the start of this study. Not only the code but the whole ecosystem created is effective. The pathfinding algorithm and the data analytics implementation give huge value to the tool, turning the analysis into a user-friendly workflow.

It is true that the results need more validation. For MEO and GEO constellations, when the orbits are similar, there are short periods where the code shows non-visibility where it is supposed to show visibility. Seems to work fine for LEO and cases where orbits are not similar.

Finally, it is worth mentioning that *Interlink* code can provide solutions for already existing constellations but also when designing a new one. Both use cases can turn this tool into a business case.

Despite the difficulties encountered, this was a project I personally enjoyed. It started with some delay at first, but with hard work and persistence the desired results eventually arrived. It is not only orbital mechanics and space mission topics that I learned, it is also a lot of coding. The experience gained during the development of the study will be useful for me and for others interested on this subject as well.

8. Environmental and security impact

Economic, social and environmental change is inherent to development. Whilst development aims to bring about positive change it can lead to conflicts. This project is an analytic study of the visibility between satellites in a constellation. For this reason, it is not impacting the environment directly. However, space industry obviously has a notorious environmental impact throughout the construction of satellites and launch, above all. This must be taken into account when designing and operating space missions.

It is worth mentioning that space debris is increasing day after day. LEO orbits are becoming crowded and this thesis provides a solution for an apparent problem affecting this kinds of orbits. This can lead to more constellations there. However, this project could also lead to better constellation designs, thus affecting the number of satellites necessary in some applications. Anyway, end-of-life disposal for each satellite sent to space is a topic that needs to be seriously faced.

It has to be mentioned that this thesis is an academical project and direct application of *Interlink* code to real cases is not recommended. Disclaimer alludes to the further validation needed. In addition, the purpose of this project is to give value to society not for malicious purposes.

9. Future work

It is clear that the work done during this project can be improved with more work and new features such as:

- Further validation.
- Better perturbations model such as SGP8 could be implemented for a better orbital decay prediction and a more accurate tracking as it takes into account solar radiation effects.
- Pathfinder algorithm could be enhanced to compute more than 3 jumps. The hypothesis adopted in this study is logic but it is not always true. There exist some configurations where the quickest path takes more than three jumps.
- Integrating a ground to satellite visibility prediction module in order to have an end to end solution.
- Translating the code to an open-source programming language such as Python.
- Spreading the tool and creating a consolidated database with all the data gathered by all users.
- Use of the gathered data to implement deep learning algorithms. This could be helpful to design better constellations in terms of visibility, giving as input the specific requirements.

Bibliography

[1] Loren Grush. FCC approves SpaceX's plan to launch more than 7,000 internet-beaming satellites, 2018. URL `https://bit.ly/2A48Tk2`.

[2] SpaceX satellite Internet project status update, 2017. URL `https://bit.ly/2VdtekA`.

[3] Jing Chu, Jian Guo, and E. K. A. Gill. Fractionated space infrastructure for long-term earth observation missions. In *EEE Aerospace Conference*, pages 1–9. IEEE, 3 2013. ISBN 978-1-4673-1813-6. doi: 10.1109/AERO.2013.6496854. URL `https://bit.ly/2H0KW0f`.

[4] Patrick W. Kelly and Riccardo Bevilacqua. Constellation Design for Mars Navigation using Small Satellites. In *AIAA Aerospace Sciences Meeting*, Reston, Virginia, 1 2018. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-524-1. doi: 10.2514/6.2018-1538. URL `https://bit.ly/2DN5id2`.

[5] Formation-flying Satellite Demo, 2013. URL `https://bit.ly/2GWRr4t`.

[6] What Is Agile Methodology?, 2017. URL `https://bit.ly/2Y23W5C`.

[7] Kanban Versus Scrum in Agile Methodology, 2017. URL `https://bit.ly/2Spmi2i`.

[8] How many satellites orbiting the Earth in 2019?, 2019. URL `https://bit.ly/2Jgk2EK`.

[9] William Cook. The Anatomy of a Satellite, 1996. URL `https://bit.ly/2LohNC9`.

[10] Mark Garcia. International Space Station Facts and Figures, 2016. URL `https://go.nasa.gov/2qU2eHz`.

[11] What is a CubeSat, 2018. URL `https://bit.ly/2Y6187I`.

[12] Virgin Orbit Completes Successful Captive Carry Test Flight, 2018. URL `https://bit.ly/2DB4tEZ`.

[13] Applications of Satellites, 2019. URL `https://bit.ly/2V0mPUN`.

[14] Kepler's Three Laws. URL `https://bit.ly/2RRxFLF`.

[15] Newton's Three Laws of Motion. URL `https://bit.ly/2GgJJB9`.

[16] FAA. Describing Orbits. Technical report. URL `https://bit.ly/2GRYQSR`.

[17] Orbital Elements, 2019. URL `https://bit.ly/2IZIf2H`.

[18] David A. Vallado and Wayne D. McClain. *Fundamentals of astrodynamics and applications*. Kluwer Academic Publishers, 2001. ISBN 9780792369035. URL `https://bit.ly/2ZPU0OA`.

[19] NASA. Catalog of Earth Satellite Orbits. 9 2009. URL `https://go.nasa.gov/2VJoz9s`.

[20] Cain Fraser. Where is the closest black hole?, 2016. URL `https://bit.ly/2VNbZG7`.

[21] Tengyue Mao, Zhengquan Xu, and Rui Hou. Efficient Constellation Design Based on Improved Non-dominated Sorting Genetic Algorithm-II. 2012. doi: 10.4304/jcp.7.6.1337-1344. URL `https://bit.ly/2J01Zn4`.

[22] Raja P. The Art of Satellite Constellation Design: What You Need to Know, 2015. URL `https://bit.ly/2JeVXOr`.

[23] Thierry Guillemin. LEO Satellite Constellations: Understanding Strengths and Limitations, 2015. URL `https://bit.ly/2PLzU39`.

[24] Lloyd Wood. Iridium as modelled in SaVi. URL `https://bit.ly/2vC904H`.

[25] Maj P Edward Chatters IV, Maj Bryan Eberhardt, and Maj S Michael Warner. Orbital Mechanics. Technical report, 2008. URL `https://bit.ly/2Sxqvxa`.

[26] Oblate spheroid archives, 2017. URL `https://bit.ly/2DON1vX`.

[27] State of Art of the Orbit Propagators, 2015. URL `https://bit.ly/2GVR2iW`.

[28] Felix R Hoots and Ronald L Roehrich. SPACETRACK REPORT NO. 3 Models for Propagation of NORAD Element Sets. Technical report, 1980. URL `https://bit.ly/2JgAAfM`.

[29] Perturbed Motion. URL `https://bit.ly/2DOwat8`.

[30] M.A. Sharaf, M.E. Awad, I.A. Hassan, R. Ghoneim, and W.N. Ahmed. Visual contact for two satellites orbits under J2-gravity. *NRIAG Journal of Astronomy and Geophysics*, 1(1):61–69, 6 2012. ISSN 20909977. doi: 10.1016/j.nrjag.2012.11.007. URL `https://bit.ly/2IZk5p3`.

[31] E H Khattab, M E Awad, M A Sharaf, and E H Khatab. Visual Contact between Two Earth's Satellites. *American Journal of Applied Sciences*, 9(5):620–623, 2012. ISSN 1546-9239. URL `https://bit.ly/2Y354pK`.

[32] Hamed Shahmohamadi Ousaloo. Magnetic Attitude Control of Dynamically Unbalanced Spinning Spacecraft during Orbit Raising. *Journal of Aerospace Engineering*, 27(2):262–278, 3 2014. ISSN 0893-1321. doi: 10.1061/(ASCE)AS.1943-5525.0000252. URL `https://bit.ly/2J4bVvT`.

[33] Two-line element set, 2019. URL `https://bit.ly/2LnTqUS`.

[34] T.S. Kelso. NORAD Two-Line Element Set Format. URL `https://bit.ly/2DqmGqe`.

[35] Meysam Mahooti. SGP4 - File Exchange - MATLAB Central, 2017. URL `https://bit.ly/2JgXjZf`.

[36] Tyler G. R. Reid. Two Line Element Reader Plotter - File Exchange - MATLAB Central, 2017. URL `https://bit.ly/2H2VJZg`.

[37] Robert Lee. Headed to Mars: A Big Experiment in Tiny Satellites, 2018. URL `https://on.wsj.com/2vDKYWO`.

[38] Bruno Luong. Plot earth - File Exchange - MATLAB Central, 2014. URL `https://bit.ly/2V2HZBB`.

[39] Echo, NASA's First Communications Satellite, 2015. URL `https://go.nasa.gov/2h6fPCB`.

[40] J. Ajala. Design of the satellite link, 2014. URL `https://bit.ly/2H5ajQ9`.

[41] Satellite frequency bands, 2018. URL `https://bit.ly/2J5hEkL`.

[42] Dough Mohney. Gigabit and faster laser communication advances for satellite use, 2018. URL `https://bit.ly/2vEQsRl`.

[43] Jennifer Chu. Laser-pointing system could help tiny satellites transmit data to Earth, 2018. URL https://bit.ly/2vDV3TF.

[44] Inter-satellite laser links, 2016. URL https://bit.ly/2LzSo8G.

[45] Omar Elgabry. Path Finding Algorithms, 2016. URL https://bit.ly/2PLU9xV.

[46] T.S. Kelso. CelesTrak, 2019. URL https://celestrak.com/.